Disseny i contrucció d'un pèndul invertit

Memòria del projecte de final de carrera corresponent als estudis d'Enginyeria en Informàtica presentat per

Xavier Jordà Múrria dirigit per **Joan Oliver**.

Bellaterra, 7 de Gener de 2013

El firmant, Joan Oliver , professor del Departament de Microelectrònica i Sistemes Electrònics de la Universitat Autònoma de Barcelona

CERTIFICA:

Que la present memòria ha sigut realitzada sota la seva direcció per Xavier Jordà Múrria

Bellaterra, 8 de Gener de 2013

Firmat: Joan Oliver

Agraïments

Vull agrair el suport i els ànims que la meva família m'ha ofert en tot moment al llarg d'aquest últim any.

També vull agrair al Dr. Joan Oliver, director d'aquest projecte, tots els consells que m'ha donat i la paciència i simpatia que ha tingut amb mi, així com al Dr. Asier Ibeas per la seva colálaboració tenint sempre la porta del seu despatx oberta per a consultes.

Finalment, vull donar les gràcies també a tots els companys i amics, que han fet més planer i agradable el pas per la universitat des de que vaig començar aquesta carrera fins avui.

Índex

1 Introducció **1**

1.1 Objectius del projecte: 2

1.2 *Software* utilitzat 2

2 Estat de l'art **5**

2.1 Anàlisis de diferents aproximacions al pèndul invertit 5

2.1.1 Exemple 1 6

2.1.2 Exemple 2 11

2.1.3 Exemple 3 17

2.2 Comparacions i conclusions de les aproximacions 22

3 Construcció i programació d'un model de pèndul invertit **23**

3.1 Parts *hardware* del prototip 23

3.1.1 Placa Arduino 23

3.1.2 ATmega328P 25

3.1.3 Xip l293D 26

3.1.4 Motors DC. 28

3.1.5 Sensor MinIMU-9 Giroscopi, Accelerometre 28

3.1.6 Sensor IR 33

3.2 Arquitectura final del pèndul invertit 33

3.3 Disseny part *software* del prototip 34

3.3.1 GUI d'Arduino 35

3.3.2 Estructura del programa basat en Arduino 36

4 Model matemàtic del pèndul invertit 37

4.1 Definició del problema: 37

4.2 Anàlisi de les forces i els sistemes d'equacions: 39

4.3 Funció de transferència: 40

4.4 Variables d'estat 41

4.5 Resposta en llaç obert del sistema 42

4.5.1 Per funció de transferència 42

4.5.2 Per variables d'estat 42

4.6 Model del motor DC 43

5 Disseny del control 47

5.1 Controlador PID 47

5.1.1 Disseny d'un controlador PID 48

5.1.2 PID aplicat en un pèndul invertit 50

5.2 Controlador LQR 55

5.2.1 Disseny d'un LQR en espai d'estats 55

6 Obtenció del senyal de control i simulacions amb diferents condicions inicials 65

6.1 Obtenció del senyal de control u 66

6.2 Simulacions amb diferents c.i. 66

6.2.1 Simulació 1: 68

6.2.2 Simulació 2: 69

6.2.3 Simulació 3: 70

6.2.4 Simulació 4: 71

6.2.5 Simulació 5: 72

6.2.6 Simulació 6: 73

6.2.7 Simulació 7: 74

6.2.8 Simulació 8: 75

6.2.9 Simulació 9: 75

7 Algoritme de control cas real **77**

7.1 Disseny de l'algoritme de control 77

7.2 Anàlisi de l'algoritme 78

7.3 Resultats 79

8 Resultats i Conclusions **81**

Bibliografia **83**

APÈNDIX **87**

A Model del motor procés Arduino **87**

B Model del motor procés Matlab **91**

C PID per a TF **93**

D Codi simulacions **95**

E Controlador cas real **97**

F LQR per a un disseny de pèndul invertit amb variables d'estat **101**

G Diagrames de Gantt **103**

Índex de figures

2.1 *Pèndul de Furuta* 6
2.2 *Disseny de control* 10
2.3 *Model del pèndul* 11
2.4 *Fragmentació del pèndul invertit* 12
2.5 *Disseny de control* 14
2.6 *Model del pèndul* 17
2.7 *Representació var. d'estat* 19
2.8 *PID* 20

3.1 *Arduino board* 24
3.2 *Microcontrolador* 25
3.3 *L293d* 26
3.4 *Motors* 27
3.5 *Motors invertits* 27
3.6 Motor DC 28
3.7 *Sensor giro/acc* 29
3.8 *Segway axis* 30
3.9 *MinIMU-9-accelerometre* 30
3.10 *Esquema de graus* 31
3.11 *Sensor IR i connexionat* 33
3.12 *Circuit del sistema* 34
3.13 *Prototip final* 34

3.14 *Consola Arduino* 35

4.1 *Pèndul invertit* 38
4.2 *H(s) en llaç obert* 42
4.3 *VVEE amb llaç obert* 43
4.4 *Potència d'entrada* vs *Velocitat de sortida* 44
4.5 *Resposta del motor* 45

5.1 *Diagrama de blocs del sistema* 48
5.2 *TF Diagrama de blocs del sistema.* 50
5.3 *TF Lloc de les arrels del sistema* 51
5.4 *TF Lloc de les arrels del sistema+PID* 52
5.5 *TF Resposta de la inclinació a una entrada impuls.* 53
5.6 *TF Diagrama de blocs del sistema+posició.* 54
5.7 *TF Resposta de la posició del carro a una entrada impuls.* 54
5.8 *LQR diagrama de blocs del sistema* 58
5.9 *Sistema amb control lqr* 59
5.10 *Diagrama de blocs+ referencia* 60
5.11 *Sistema amb el control lqr+ referencia* 60
5.12 *Simulació de l'observador* 61
5.13 *Simulació observador* 62

6.1 *Simulació 1* 68
6.2 *Simulació 2* 69
6.3 *Simulació 3* 70
6.4 *Simulació 4* 71
6.5 *Simulació 5* 72
6.6 *Simulació 6* 73
6.7 *Simulació 7* 74
6.8 *Simulació 8* 75
6.9 *Simulació 9* 76

7.1 *Fotogrames de la demostració* 79

G.1 *Diagrama de Gantt planificació inicial* 103
G.2 *Diagrama de Gantt reunions director* 104
G.3 *Diagrama de Gantt planificació final* 106
G.4 *Total hores dedicades* 106

.

Capítol 1

Introducció

El pèndul invertit és un problema clàssic en l'enginyeria de control, el qual consisteix en una varilla amb una massa en un extrem, i en l'altre extrem un eix que pot pivotar bidimensionalment sobre un carro. Aquesta, és una de les formes més usuals en què trobem representat el pèndul invertit, però n'hi ha d'altres com per exemple un carro amb dues rodes on la pròpia base del carro pivota sobre l'eix de les rodes tipus "segway".

L'interès en l'estudi d'aquests sistemes resideix en que, excloent les particularitats de cada cas, el seu model matemàtic mostra pocs punts d'estabilitat local i es tracta amb una formulació basada en equacions diferencials. El model té certes similituds amb processos reals de gran complexitat, com per exemple: un generador sincro connectat a un bus infinit o un sistema de control de vol per aeronaus.

Els primers pènduls invertits foren construïts al principi dels anys 70, tot i que 40 anys després es segueix experiementant amb ells, sobretot com a exemple pràctic en les classes de teoria de control.

La dificultat del control del pèndul invertit pot ser dividida en dos processos principals:

- 1r: L'estabilitat local al voltant de la posició d'equilibri. Aquest problema pot ser resolt mitjançant la linealització al voltant de la posició d'equilibri, ja que en aquest cas el model d'un pèndul invertit es descriu per un sistema no lineal. D'altra banda, la llei de control obtinguda un cop linealitzat té caràcter local. Per tant, per pertorbacions de magnitud elevada, el control aplicat al pèndul es perdrà.

- 2n: El problema del *swing up* consisteix en aixecar el pèndul des d'una posició de repòs

fins a una vertical propera al punt d'equilibri. Per resoldre aquest problema és necessari considerar tot l'espai d'estats impossibilitant la linealització del sistema i tractar-lo d'una forma local.

Així doncs, per tal de resoldre el problema de control del pèndul invertit, convé ressaltar que no existeix un mètode universal per fer-ho, sinó que aquest depèn de cada cas en particular. Davant aquest repte, hom ha d'utilitzar tot el seu coneixement per tal de combinar diverses metodologies per resoldre els diferents aspectes que presenta, un procés aparentment simple però que, en realitat, és molt complex degut a la seva forma particular de comportament.

1.1 Objectius del projecte:

L'objectiu d'aquest projecte és el de dissenyar un controlador per estabilitzar un prototip de pèndul invertit, en una posició vertical, sota el control del posicionament del carro. Per tant, podem dividir el treball en dos objectius pincipals:

- 1r: Construcció d'un prototip de pèndul invertit.
  - Dissenyar un prototip de pèndul invertit (de cost reduit).
  - Seleccionar els sensors, actuadors i un microcontrolador més adient.
  - Assemblar totes les peces per l'obtenció del prototip.
  - Dissenyar i implementar un *software* capaç d'interaccionar amb tots els elements anteriorment connectats d'una forma correcta.
- 2n: Controlar el sistema
  - Obtenir un model matemàtic a partir de les lleis físiques que descriguin el comportament del sistema.
  - Dissenyar i simular diferents tipus de control.
  - Implementar un controlador.

1.2 *Software* utilitzat

Aquesta memòria ha estat escrita amb l'editor de text Latex el TexShop. El microcontrolador utilitzat ha estat programat amb el seu propi *software* Arduino i amb C++. També s'ha utilitzat

l'AVR Simulator per poder observar amb més detall totes les instruccions a baix nivell que genera l'algoritme de control. Finalment tots els càlculs matemàtics i simulacions han estat realitzades amb Matlab.

Capítol 2

Estat de l'art

Avui en dia es coneixen una gran varietat de pènduls invertits. Tots ells amb les seves variants de comportament amb formes diferents d'extreure el model matemàtic, i sovint poc explícits en la forma d'obtenció de les equacions de comportament.
En aquest capítol es mostren 3 exemples de pèndul invertit, amb els seus respectius mètodes emprats per aconseguir el model matemàtic i el disseny final de control.

2.1 Anàlisis de diferents aproximacions al pèndul invertit

Al llarg dels següents exemples, es mostren les diferents estratègies seguides per aconseguir el model matemàtic i el disseny de control aplicat en cada cas concret de pèndul invertit. En conseqüència, s'extraurà la millor estratègia a seguir per al disseny del control per al prototip.

2.1.1 Exemple 1

‘‘ DISEÑO, CONSTRUCCIÓN Y CONTROL DE UN PÉNDULO INVERTIDO ROTACIONAL UTILIZANDO TÉCNICAS LINEALES Y NO LINEALES. ’’ (Carlos Andrés Osorio Ziga)

El pèndul de Furuta, que es mostra a la figura 2.1, consta de dos cossos inercials connectats: d'una banda, un pilar central amb moment d'inèrcia "J", rígidament connectat amb un braç horitzontal "l_a" i massa homogèniament distribuïda en l'eix "m_a", i de l'altra, el pèndul de longitud "l_p" i massa homogèniament distribuïda en l'eix "m_p". L'angle del pèndul "$Theta(\Theta)$" ha estat definit com 0 en la posició vertical, mentre que l'angle del braç "$Phi(\Phi)$" ha estat definit com a positiu quan el braç es mou en la direcció contrària a les manetes del rellotge.

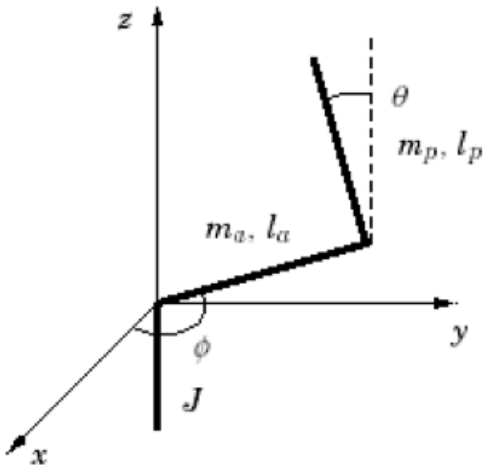


Figura 2.1: *Pèndul de Furuta*

J: moment d'inèrcia de l'eix central.

la: longitud braç horitzontal.

ma: massa homogèniament distribuïda en línia del braç horitzontal.

lp: longitud del pèndul.

mp: massa del pèndul homogèniament distribuïda en línia.

θ: l'angle del pèndul ha estat definit com 0 en la posició de 90° amb el braç horitzontal.

ϕ: l'angle del braç horitzontal és positiu quan va en direcció contrària a les manetes del rellotge.

L'obtenció del model matemàtic del sistema es basa en la teoria d'Euler-Lagrange.

Cinemàtica:

La **posició** d'un punt *P(r_x,r_y,r_x)* sobre el pèndul pot ser descrit pel següent vector:

r_a: Posició radial del braç.

r_p: Posició radial del pèndul.

$$r(r_a, r_p) = (r_x(r_a, r_p), r_y(r_a, r_p), r_z(r_a, r_p))^\tau \tag{2.1}$$

on,

$$\begin{aligned} r_x(r_a, r_p) &= r_a \cdot \cos\phi - r_p \cdot \sin\phi \cdot \sin\theta \\ r_y(r_a, r_p) &= r_a \cdot \sin\phi + r_p \cdot \sin\theta \cdot \cos\phi \\ r_z(r_a, r_p) &= r_p \cdot \cos\theta \end{aligned} \tag{2.2}$$

La **velocitat** sobre un punt *P(v_x,v_y,v_z)* s'obté prenent la derivada sobre el temps de (2.1).

$$v(r_a, r_p) = (v_x(r_a, v_p), v_y(r_a, r_p), v_z(r_a, r_p))^\tau \tag{2.3}$$

d'un punt P sobre el pèndul.

$$\begin{aligned} v_x(r_a, r_p) &= -r_a \cdot \sin\phi \cdot \dot{\phi} - r_p \cdot \cos\theta \cdot \sin\phi - r_p \cdot \sin\theta \cdot \cos\phi \cdot \dot{\phi} \\ v_y(r_a, r_p) &= r_a \cdot \cos\phi \cdot \dot{\phi} + r_p \cdot \cos\theta \cdot \cos\phi \cdot \theta - r_p \cdot \sin\theta \cdot \sin\phi \cdot \dot{\phi} \\ v_z(r_a, r_p) &= -r_p \cdot \sin\theta \cdot \dot{\theta} \end{aligned} \tag{2.4}$$

Equació utilitzada per expressar el quadrat de la magnitud de la velocitat de P.

$$v^2(r_a, r_p) = (r_a^2 + r_p^2 \cdot \sin^2\theta) \cdot \dot{\phi}^2 + 2 \cdot r_a \cdot r_p \cdot \cos\theta \cdot \dot{\phi} \cdot \dot{\theta} + r_p^2 \cdot \dot{\theta}^2 \tag{2.5}$$

Expresions d'energia:

L'energia total és la suma de les seves parts.

Energia cinètica total: $T{=}T_c + T_k + T_p$

Energia potencial total: $V{=}V_c + V_k + V_p$

Pilar central:

$$2 \cdot T_c = J \cdot \dot{\varphi}^2 \tag{2.6}$$

$$V_c = 0 \tag{2.7}$$

Braç horitzontal

$$2T_c = \frac{1}{3} \cdot m_a \cdot l_a^2 \cdot \dot{\varphi}^2 \tag{2.8}$$

$$V_a = 0 \tag{2.9}$$

Braç pendular:

$$2 \cdot T_p = m_p \cdot (l_a^2 + \frac{1}{3} \cdot l_p^2 \cdot \sin^2\theta) \cdot \dot{\varphi}^2 + m_p \cdot l_a \cdot l_p \cdot \cos\theta \cdot \dot{\varphi} \cdot \dot{\theta} + \frac{1}{3} \cdot m_p \cdot l_p^2 \cdot \dot{\theta}^2 \tag{2.10}$$

$$V_p = \frac{1}{2} \cdot m_p \cdot g \cdot l_p \cdot \cos\theta \tag{2.11}$$

Equacions de moviment

Formulant el Lagrangià L= T-V les equacions de moviment estan donades per:

$$\begin{aligned} \frac{d}{dt}\left(\frac{\delta L}{\delta \dot{\phi}}\right) - \frac{\delta L}{\delta \phi} &= \tau_\phi \\ \frac{d}{dt}\left(\frac{\delta L}{\delta \dot{\theta}}\right) - \frac{\delta L}{\delta \theta} &= 0 \end{aligned} \tag{2.12}$$

Sent τ_ϕ una força externa aplicada al braç horitzontal, les derivades parcials són:

$$\begin{aligned} \frac{\delta L}{\delta \phi} &= 0 \\ \frac{\delta L}{\delta \dot{\phi}} &= \left(J + \left(\frac{1}{3} \cdot m_a + m_p\right) \cdot l_a^2 + \frac{1}{3} \cdot m_p \cdot l_p^2 \cdot \sin^2\theta\right) \cdot \dot{\phi} + \frac{1}{2} \cdot m_p \cdot l_p \cdot l_a \cdot \cos\theta \cdot \dot{\theta} \\ \frac{\delta L}{\delta \theta} &= \frac{1}{3} \cdot m_p \cdot l_p^2 \cdot \cos\theta \cdot \sin\theta \cdot \dot{\phi}^2 - \frac{1}{2} \cdot m_p \cdot l_a \cdot l_p \sin\theta \cdot \dot{\phi} \cdot \dot{\theta} + \frac{1}{2} \cdot m_p \cdot g \cdot l_p \cdot \sin\theta \\ \frac{\delta L}{\delta \dot{\theta}} &= \frac{1}{2} \cdot m_p \cdot l_a \cdot l_p \cdot \cos\theta \cdot \dot{\phi} + \frac{1}{3} \cdot m_p \cdot l_p^2 \cdot \dot{\theta} \end{aligned} \tag{2.13}$$

Inserint (2.13) en (2.12) i introduïnt les següents variables

$$\begin{aligned} \alpha &= J + (13 \cdot m_a + m_p)_a^2 & \beta &= \frac{1}{3} \cdot m_p \cdot l_p^2 \\ \gamma &= \frac{1}{2} \cdot m_p \cdot l_a \cdot l_p & \delta &= \frac{1}{2} \cdot m_p \cdot g \cdot l_p \end{aligned} \tag{2.14}$$

obtenim les equacions de moviment per al sistema:

$$\begin{aligned}\left(\alpha+\beta\cdot\sin^2\theta\right)\cdot\ddot{\phi}+\gamma\cdot\cos\theta\cdot\ddot{\theta}+2\cdot\beta\cdot\cos\theta\cdot\sin\theta\cdot\dot{\phi}\cdot\dot{\theta}-\gamma\cdot\sin\theta\cdot\dot{\theta}^2=\tau_\phi\\ \gamma\cdot\cos\theta\cdot\ddot{\phi}+\beta\cdot\ddot{\theta}-\beta\cdot\cos\theta\cdot\sin\theta\cdot\dot{\phi}^2-\delta\cdot\sin\theta=0\end{aligned}\tag{2.15}$$

Representació en espai d'estats:

Un cop substituïdes les diferents variables d'estat $x_1=\phi$, $x_2=\dot{\phi}$, $x_3=\theta$, $x_4=\dot{\theta}$, les equacions de moviment anteriors es poden descriure d'una forma més entenedora i molt més apropiada per la integració i disseny de control en espai d'estats.

$$\begin{aligned}\dot{x_1} &= x_2\\ \dot{x_2} &= \frac{\beta\cdot\gamma\left(\sin^2\theta-1\right)\cdot\sin\theta\cdot\dot{\varphi}^2-2\cdot\beta^2\cdot\cos\theta\cdot\sin\theta\cdot\dot{\varphi}\cdot\dot{\theta}-\gamma\cdot\delta\cdot\cos\theta\cdot\sin\theta+\beta\cdot\tau_\varphi}{\alpha\cdot\beta-\gamma^2+(\beta^2+\gamma^2)\cdot\sin^2\theta}\\ \dot{x_3} &= x_4\\ rest &= \gamma^2\cdot\cos\theta\sin\theta\cdot\dot{\theta}^2+\delta\cdot\left(\alpha\cdot\sin^2\theta\right)\cdot\sin\theta-\gamma\cdot\cos\theta\cdot\tau_\varphi\\ \dot{x_4} &= \frac{\beta\cdot\left(\alpha+\beta\cdot\sin^2\theta\right)\cdot\cos\theta\cdot\sin\theta\cdot\dot{\varphi}^2+2\cdot\beta\cdot\gamma\left(1-\sin^2\theta\right)\cdot\sin\theta\cdot\dot{\varphi}\cdot\dot{\theta}-rest}{\alpha\cdot\beta-\gamma^2+(\beta^2+\gamma^2)\cdot\sin^2\theta}\end{aligned}\tag{2.16}$$

Control del pèndul invertit

Per tal de simplificar el disseny comentat anteriorment, s'utilitza un model simplificat del pèndul de Furuta. Aquest menysprea els moments de força de reacció exercits des del pèndul cap al braç. Així doncs, el model matemàtic obtingut és de 2n ordre. Les tècniques de control aplicades són les següents:

- Controlador *swing up* mitjançant un control de l'energia aplicada al sistema. Control que aproxima el pèndul a l'origen, però mai arribarà a passar per aquest.
- Control lineal (LQR), ja que tots els estats són mesurables per realimentació del sistema, amb la finalitat d'estabilitzar localment el punt d'equilibri (0,0,0,0) (durant el següent exemple es realitza una definició més precisa del control LQR).
- Utilització d'un precompensador amb acció integral (PI), ja que aquest sistema de control no considera directament les especificacions estàtiques.

D'aquesta manera, el control final del sistema està format per un precompensador *PI* i un controlador híbrid (*swing up*+LQR).
En la següent figura s'observa el diagrama de blocs final del control d'aquest sistema.

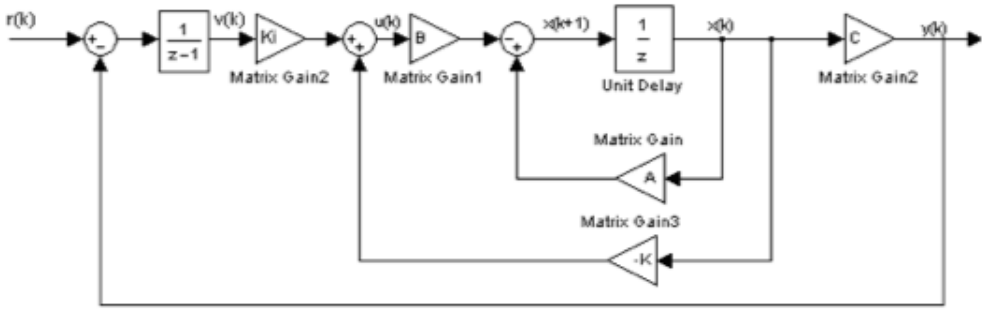


Figura 2.2: *Disseny de control*

2.1.2 Exemple 2

‘‘ LEVANTAMIENTO Y ESTABILIZACIÓN DEL PÉNDULO INVERTIDO. ’’ (Fernando Castaos Luna)

Aquest model de pèndul invertit consta de dues parts: una petita vagoneta que està damunt d'una cinta que li fa de carril, de tal manera que només es pot desplaçar cap endavant i cap endarrere, i el pèndul invertit en si, que està a sobre de la vagoneta.

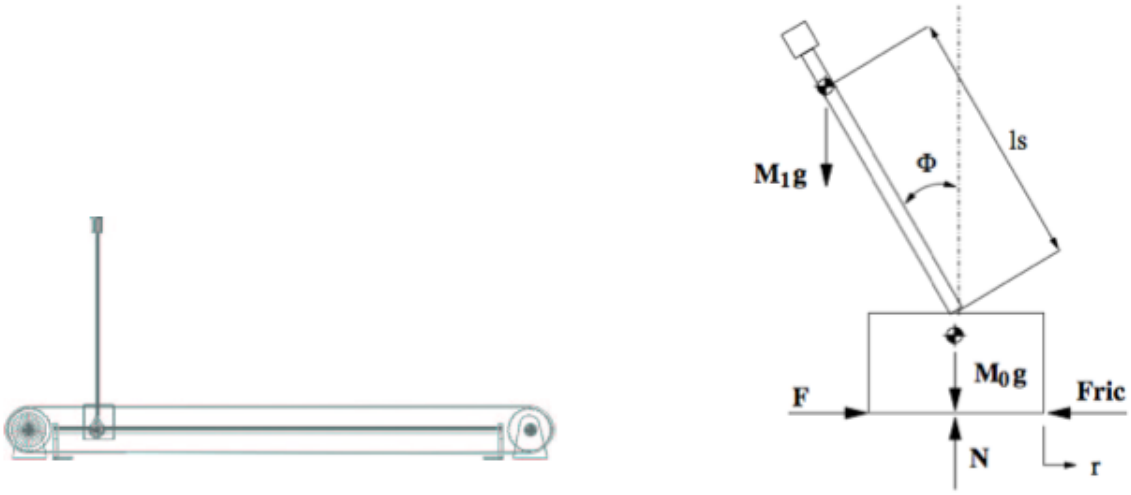


Figura 2.3: *Model del pèndul*

Equacions diferencials

$$\sum_{i=1}^{+\infty} F_i = m \cdot a_i \tag{2.17}$$

$$\sum_{j=1}^{+\infty} F_j = m \cdot a_j \tag{2.18}$$

$$\sum_{i=1}^{+\infty} M_g = m \cdot \alpha_i \tag{2.19}$$

Aquest sistema es pot entendre com un cos rígid on el seu moviment està limitat a dues dimensions.

Les equacions (2.17) i (2.18) són l'aplicació de la 2a llei de Newton pels components horitzontals(i) i verticals(j) de la força "F" i l'acceleració experimentada pel cos rígid de massa " m ".

L'equació (2.19) ens mostra que la suma de moments "M" sobre un punt "g" és igual al moment d'inèrcia "I" per la velocitat angular "α" direcció "g".

Equacions de moviment

Aplicant les equacions (2.17) , (2.18) i (2.19) al pèndul invertit s'obtenen les equacions que defineixen el moviment d'aquest.

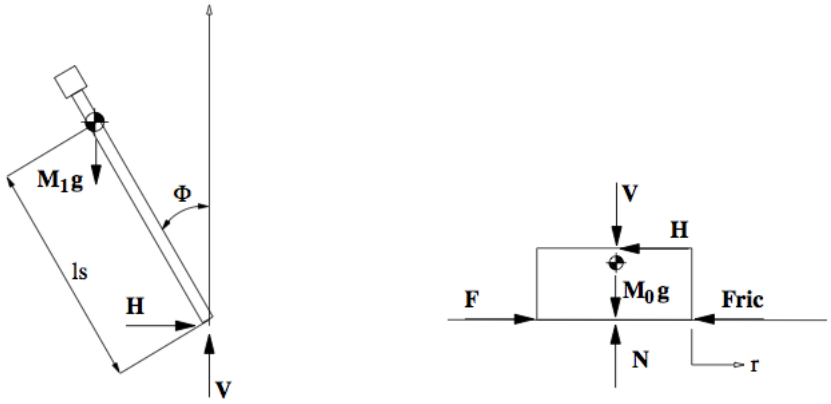


Figura 2.4: *Fragmentació del pèndul invertit*

En la figura 2.4 s'observa que existeixen 3 moments de força diferents: la força horitzontal "h", la vertical "v" i un tercer a causa de la resistència de fricció que produeix l'articulació.

$$h = M_1 \cdot \left(\ddot{r} - l_s \cdot \ddot{\Phi} \cdot \cos\Phi + l_s \cdot \dot{\Phi}^2 \cdot \sin\Phi\right) \tag{2.20}$$

$$v = M_1 \cdot l_s \left(-\ddot{\Phi}\sin\Phi - \dot{\Phi}^2 \cdot \cos\Phi\right) + M_1 \tag{2.21}$$

$$\Theta_s \frac{d^2\Theta}{dt^2} = v \cdot l_s \cdot \sin\Phi + h \cdot l_s \cos\Phi - C \cdot \frac{d\Phi}{dt} \tag{2.22}$$

Analitzant el diagrama del cos lliure del carro s'obté:

$$F - Fric - h = M_0 \cdot \frac{d^2r}{dt} \tag{2.23}$$

Substituint (2.20) i (2.21) a (2.22)

$$\Theta \cdot \ddot{\Phi} + C \cdot \dot{\Phi} - M_1 \cdot l_s \cdot (\ddot{r} \cdot \cos\Phi + g \cdot \sin\Phi) = 0 \tag{2.24}$$

Per resoldre aquesta equació diferencial amb dues incògnites es precisa d'una equació addicional que les relacioni (2.23).

F_r = constant de proporcionalitat.
$M = M_0 + M_1$

$$M \cdot \ddot{r} + F_r \cdot \dot{r} + M_1 \cdot l_s \left(\dot{\Phi}^2 \cdot \sin \Phi - \ddot{\Phi} \cdot \cos \Phi \right) = F \tag{2.25}$$

Les equacions (2.24) i (2.25) representen un sistema de dos equacions diferencials simultànies amb dues incògnites que descriuen el pèndul invertit.

Representació en espai d'estats:
Qualsevol conjunt d'equacions diferencials lineals oridinàries es pot representar com un conjunt d'equacions de primer ordre. Aquesta representació és coneguda com a equacions d'estat, i consta de la següent estructura:

$$\begin{aligned} \dot{x} &= A \cdot x + B \cdot u \\ y &= C \cdot x + D \cdot u \end{aligned} \tag{2.26}$$

x⇒ estat del sistema, i conté n elements.

u⇒ vector d'entrada, i conté m elements.

y⇒ vector de sortida, i conté p elements.

A⇒ matriu del sistema n*n.

B⇒ matriu d'entrada n*m.

C⇒ matriu de sortida p*n.

D⇒ matriu de dimensió p*m.

$$x = \begin{bmatrix} r \\ \Phi \\ \dot{r} \\ \dot{\Phi} \end{bmatrix}; \qquad y = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}; \qquad u = F \tag{2.27}$$

Les entrades del sistema són els valors obtinguts pels sensors de què disposa el pèndul, més l'observació d'una que no es mesura. Els 3 sensors de què disposa són: un per "r", un per "Φ" i un altre per "$\dot{r}$".

La sortida del sistema està definida pel vector "y". Finalment el vector "u" consta d'un sol element que és la força "F" aplicada al carro.

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{\alpha^2 \cdot g}{\beta} & -\frac{\Theta_r}{\beta} & -\frac{\alpha \cdot C}{\beta} \\ 0 & \frac{\alpha \cdot M \cdot g}{\beta} & -\frac{\alpha \cdot F_r}{\beta} & -\frac{M}{\beta} \end{bmatrix}; \qquad B = \begin{bmatrix} 0 \\ 0 \\ \frac{\Theta}{\beta} \\ \frac{\alpha}{\beta} \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \qquad D = 0 \tag{2.28}$$

Control del pèndul invertit

Es proposa un disseny retroalimentat de l'estat a través de d'un mètode LQR (*Linear Quadratic Regulator*).

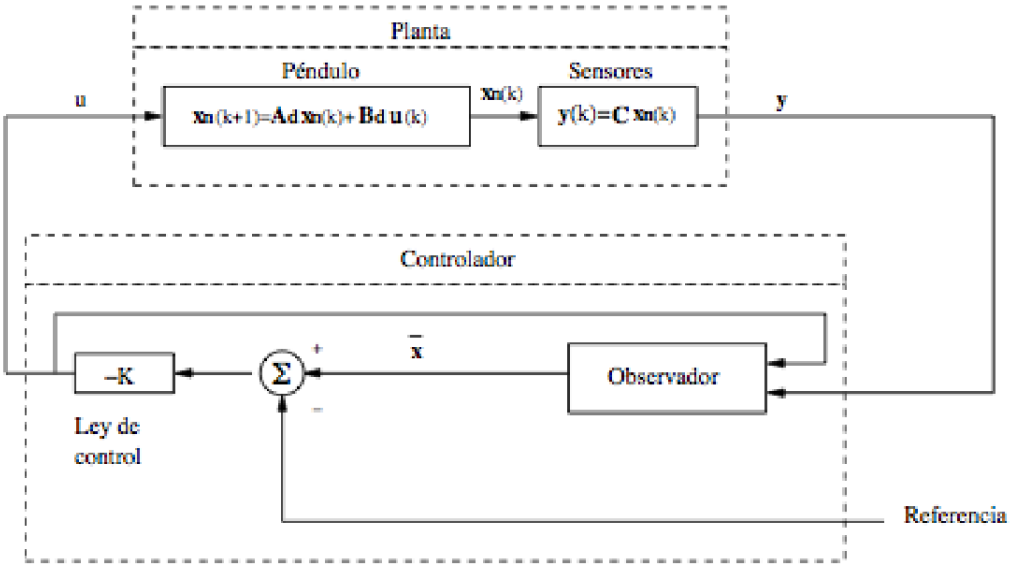


Figura 2.5: *Disseny de control*

Base teòrica d'un controlador LQR

Les tècniques de control òptim formen part d'una de les branques de control automàtic més importants en el desenvolupament d'estratègies modernes de control d'avui en dia.
Les estratègies de control òptim calculen la llei de control, de manera que amb una mesura concreta s'optimitzi el controlador.
El sistema descrit inicialment mitjançant equacions d'estat és:

$$\dot{x} = Ax + Bu$$

L'objectiu és trobar una llei de control u(k), tal que:

$$u(k) = -Kx(k)$$

on la matriu K[1] té com a finalitat minimitzar la funció cost J (que expressa un índex de funcionament)

$$J = \frac{1}{2}\sum_{k=0}^{\infty}(x^T(k)Qx(k) + u^T(k)Ru(k)) \tag{2.29}$$

on Q i R són matrius de ponderació, les quals compleixen que $Q^T = Q > 0$ i $R^T = R > 0$. Cal destacar que l'índex de funcionament pondera la diferència entre l'estat actual i l'origen neutral, en un instant inicial fins a un temps infinit. Per tant, com més ràpid s'arribi a l'origen més petit serà el valor de J. Això implica que al minimitzar J, hom obtindrà la llei de control que porta l'estat a l'origen d'una manera més ràpida. D'altra banda, s'observa que en la funció cost hi ha un terme que impedeix que es porti l'estat a l'origen, a expences d'una actuació molt gran. Al minimitzar J, el que es pretén és trobar una solució de compromís entre el rendiment del controlador i el seu nivell d'actuació. La raó d'ésser d'aquest compromís pot venir donada per diferents motius, com ara reduir el consum d'energia necessària per a proporcionar el senyal d'actuació. Tantmateix, existeixen altres motius i no menys importants, com poden ser les possibles discrepàncies entre el model del sistema i la seva dinàmica real això passa quasi sempre, aquesta ponderació del qual resulta en un sistema més estable.
Així doncs, per tal de poder calcular la llei de control que minimitzi la funció cost 2.29 fou definida una matriu P que satisfaci la següent equació, anomenada de Riccatti.

$$P = Q + G^T PG - G^T PH(R + H^T PH)^{-1}H^T PG \tag{2.30}$$

[1]la matriu K, és la matriu que defineix la llei de control del controlador LQR i que es troba definida en la següent equació 2.31.

La solució d'aquesta equació és una matriu P, que és hermítica[1] i definida positiva, demostrant que la matriu K

$$K = (R + H^T PH)^{-1} H^T PG \tag{2.31}$$

és la que minimiza la funció cost 2.29 a través de la llei de control

$$u(k) = -(R + H^T PH)^{-1} H^T PGx(k)$$

El valor exacte de J no és rellevant, el que pretén el mètode és trobar una K que asseguri que aquest valor és mínim. L'important és el valor relatiu que tenen els elements Q i R entre si. Aquests paràmetres seran els encarregats de balancejar la importància relativa de l'entrada i els estats en la funció de cost que s'està intentant optimitzar. El cas més simple és considerar R=1 i proposar un valor de Q com a punt d'inici.

[1] Una matriu hermítica és una matriu quadrada d'elements complexos que té la característica de ser igual a la seva pròpia transposda conjugada.

2.1.3 Exemple 3

‘‘ CONSTRUCCIÓN Y CONTROL DE UN PÉNDULO INVERTIDO UTILIZANDO LA PLATAFORMA LEGO MINDSTORMS NXT. ’’
(Ing. Hctor Snchez, Ph.D. Iaki Aguirre, Ph.D. Anna Patete)

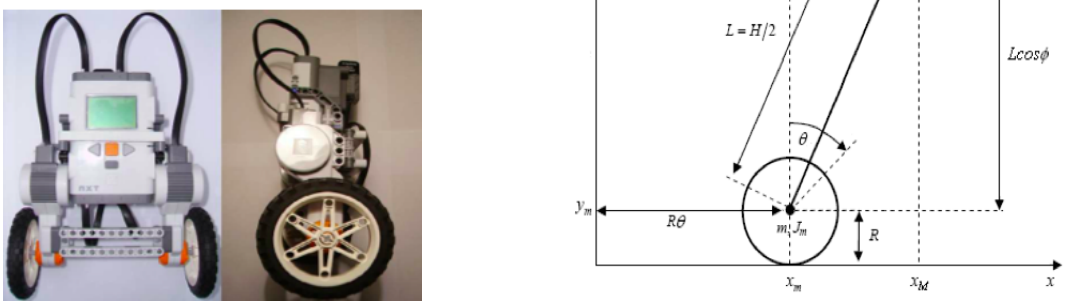


Figura 2.6: *Model del pèndul*

L= Distància entre el centre de massa i l'eix de les rodes.

H= Altura del cos del pèndul.

M= Massa del cos del pèndul.

R= Radi de les rodes.

J_m= Moment d'inèrcia de les rodes.

J_Φ= Moment d'inèrcia del pèndul.

g= Acceleració de la gravetat.

Φ= Angle d'inclinació del cos del pèndul.

Θ= Angle de rotació de les rodes.

m= Massa de la roda.

Aquest model de pèndul invertit consta d'un pèndul a sobre d'una plataforma que es desplaça amb dues rodes. Aquest model és el que més s'aproxima al model proposat en aquest treball. Per tal de trobar el model funcional que defineix el moviment del sistema, aquest exemple fa servir el model d'energia composat per l'energia cinètica total i l'energia potencial de l'objecte.

Expressions d'energia:

- Energia cinètica translacional:

$$T_1 = R \cdot \dot{\Theta}^2 + \frac{1}{2} \cdot M \cdot \left(R \cdot \dot{\Theta} + L \cdot \dot{\phi} \cdot \cos\phi \right)^2 + \frac{1}{2} \cdot M \cdot \left(L \cdot (-\sin\phi) \cdot \dot{\phi} \right)^2 \tag{2.32}$$

- Energia cinètica rotacional:

$$T_2 = J_m \cdot \dot{\Theta}^2 + \frac{1}{2} \cdot J_\Phi \cdot \dot{\Phi}^2 + \frac{1}{2} \cdot M \cdot \dot{Y_m}^2 \tag{2.33}$$

- Energia potencial:

$$U = 2 \cdot m \cdot \ g \cdot R + M \cdot g \cdot \ (R + L \cdot \cos\Phi) \tag{2.34}$$

Equacions de moviment:

Com ja s'ha pogut veure en el 1r exemple d'aquest capítol, el Lagrangià és la suma de les energies translacional i rotacional, menys l'energia potencial:

(2.35)

$$L_{ag} = T_1 + T_2 - U \tag{2.36}$$

Les equacions de Lagrange són:

$$\frac{d}{dt} \cdot \left(\frac{\delta L_{ag}}{\delta \dot{\Theta}}\right) - \frac{\delta L_{ag}}{\delta \Theta} = F_\Theta \tag{2.37}$$

$$\frac{d}{dt} \cdot \left(\frac{\delta L_{ag}}{\delta \dot{\Phi}}\right) - \frac{\delta L_{ag}}{\delta \Phi} = F_\Phi \tag{2.38}$$

Substituint l'equació (2.35) en les equacions (2.37) i (2.38) obtenim les equacions de moviment de Lagrange per al pèndul invertit sobre dues rodes.
Prenent com a referència el model del pèndul invertit sobre un carro realitzat per Ogata ref.[5], s'escull com a força externa el voltatge aplicat a les rodes, el qual és el senyal de control U. Per tant, diem que $F_\Theta = U$ i $F_\Phi = 0$, ja que considerarem que inicialment l'angle d'inclinació del cos del pèndul està en repòs.

Aplicant aquestes modificacions a les equacions del moviment de Lagrange i aïllant, obtenim les següents equacions:

$$\ddot{\Phi} = \frac{-(M\cdot L\cdot R\cdot \cos\Phi)\cdot(U + M\cdot L\cdot R\cdot \dot{\Phi}^2\cdot \sin\Phi) + M\cdot g\cdot L\cdot \sin\Phi[(2\cdot m + M)\cdot R^2 + 2\cdot J_m]}{[(2\cdot m + M)\cdot R^2 + 2\cdot J_m]\cdot(M\cdot L^2\cdot J_\Phi) - (M\cdot L\cdot\ R\cdot \cos\Phi)^2} \tag{2.39}$$

$$\ddot{\theta} = \frac{(M\cdot L^2 + J_\Phi)\cdot(U + M\cdot L\cdot R\cdot \sin\Phi\cdot\dot{\Phi}^2) - (M\cdot L)^2\cdot R\cdot g\cdot \sin\Phi\cdot\cos\Phi}{[(2\cdot m + M)\cdot R^2 + 2\cdot J_m]\cdot(M\cdot L^2\cdot J_\Phi) - (M\cdot L\cdot\ R\cdot \cos\Phi)^2} \tag{2.40}$$

Representació en variables d'estat:

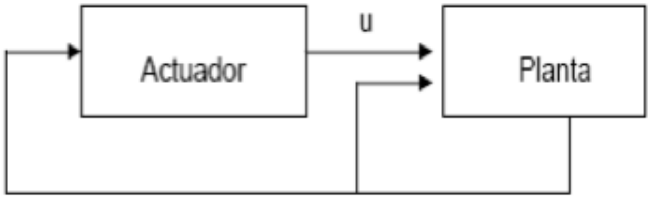


Figura 2.7: *Representació var. d'estat*

Les variables d'estat $\dot{X}_1$ i $\dot{X}_2$ representen respectivament la velocitat i l'acceleració del cos del pèndul invertit sobre dues rodes, mostrant la dinàmica de la planta.
La dinàmica de control es realitza de la següent forma: la planta té uns sensors de velocitat i rotació, aquestes dades van a parar a l'actuador, que decideix quin senyal de control (u) ha d'aplicar, per enviar-lo seguidament a la planta.

$$\dot{X}_1 = X_2 \qquad \dot{X}_2 = \frac{-M\cdot L\cdot R\cdot \cos X_1}{M\cdot L^2\cdot J_\Phi}\cdot V + \frac{M\cdot g\cdot L\sin X_1}{M\cdot L^2\cdot J_\Phi} \tag{2.41}$$

Per angles petits s'apliquen les següents transformacions: $\cos x = 1, \qquad \sin x = x.$
Llavors el nou sistema de 2n ordre queda representat per:

$$\begin{bmatrix}\dot{X_{1\delta}}\\ \dot{X_{2\delta}}\end{bmatrix} = \begin{bmatrix}0 & 1\\ \frac{M\cdot g\cdot L}{M\cdot L^2+J_\Phi} & 0\end{bmatrix}\cdot\begin{bmatrix}X_{1\delta}\\ X_{2\delta}\end{bmatrix} + \begin{bmatrix}0\\ \frac{-M\cdot L\cdot R}{M\cdot L^2+J_\Phi}\end{bmatrix}\cdot U_\delta \tag{2.42}$$

$$Y = \begin{bmatrix}1 & 0\end{bmatrix}\cdot X_\delta \tag{2.43}$$

La variable de sortida $X_{1\delta}$ és l'angle del cos del pèndul i s'obté integrant numèricament $X_{2\delta}$, que és la velocitat angular del cos del pèndul.

Disseny del control del péndul invertit :

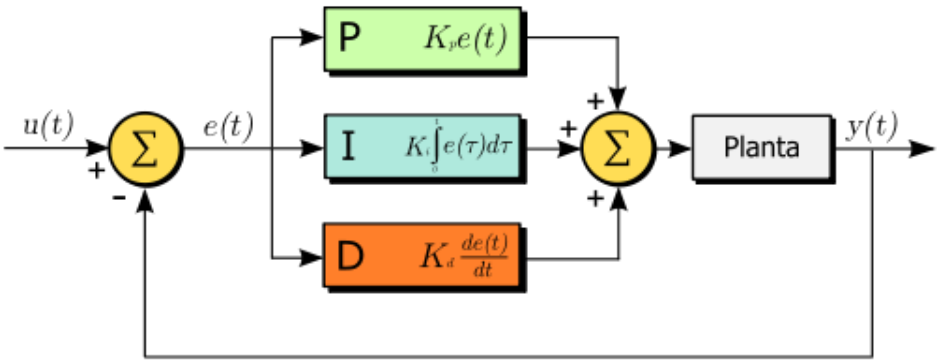


Figura 2.8: *PID*

El control proposat en aquest cas és un controlador PID.
Procés de disseny d'un PID:

- Aconseguir la funció $G(s)$ a partir de la seva representació de les variables d'estat.
- Marcar el pols que ens garanteixi els objectius del sistema.
- Extreure tots els valors necessaris per a la realització del controlador PID, a partir dels pols anteriorment comentat.

Els tres components d'un controlador PID són: part proporcional, part derivativa i part integral. El pes de la influència de cadascuna d'aquestes parts en la suma final ve donada per una constant proporcional K_p, un temps integral T_i i un temps derivatiu T_d, depenent de cadascuna de les parts observades.

- Part Proporcional: La part proporcional consisteix en el producte d'un senyal d'error $e_{rror}(t)$ per una constant proporcional K_p, amb l'objectiu que l'error en l'estat estacionari sigui pràcticament nul. La part proporcional no considera el temps, per tant, la millor manera que l'error en el permanent sigui nul és utilitzant accions integrals i derivatives.
  La fòrmula de la part proporcional ve donada per: $Part_{pro} = K_p \cdot e_{rror}(t)$

- Part Integral: La part integral té com a propòsit disminuir i eliminar l'error de l'estat estacionari, provocat per la part proporcional. El control integral actua quan hi ha una desviació entre la variable i la consigna, integrant aquesta desviació en el temps i sumant-la a l'acció proporcional. La resposta integral és addicional a la part proporcional, formant així un control P+I.
  La fórmula de la part integral és: $Part_{inte} = \int_0^t e_{rror}(t)dt$

- Part Derivativa: L'acció derivativa es manifesta quan hi ha un canvi en el valor absolut de l'error (si l'error és constant només actuen la part proporcional i integral). La funció de la part derivativa és mantenir l'error proporcional al mínim, corregint-lo proporcionalment en la mateixa velocitat que es produeix, per evitar d'aquesta manera l'increment de l'error. És important destacar que a major part derivativa es correspon un canvi més ràpid, i que s'ha de tenir en compte en el moment de triar el tipus de control desitjat.
  La fórmula de la part derivativa és: $Part_{der} = K_D \cdot \frac{de_{rror}}{dt}$

Per tant, l'equació que descriu el controlador PID és $u(t)$:

$$\begin{aligned} K_I &= \frac{K_p}{T_I} \\ K_D &= K_P \cdot T_D \\ u(t) &= K_P \cdot e_{rror}(t) + K_I \cdot \int_0^t e_{rror}(t)dt \ + K_D \cdot \frac{de_{error}}{dt} \\ error &= ref - y \end{aligned} \tag{2.44}$$

El càlcul dels paràmetres del controlador PID(K_p,T_I,T_D) segueix el protocol de l'exemple que es troba en el capítol 5, apartat 5.1.1: *Disseny d'un controlador PID.*

2.2 Comparacions i conclusions de les aproximacions

En aquesta secció es resumeixen els trets més característics –model del sistema, representació i disseny del control– dels exemples plantejats anteriorment.

Exemple 1

Model del sistema: Utilitza la teoria de Lagrange basant-se en la diferència entre les energies cinètiques i potencial.
Representació: Variables d'estat.
Tipus de control: Dos controladors, *swing up+LQR* i un *PI*.

Exemple 2

Model del sistema: Utilitza la 2a llei de Newton, que es basa en la suma de les forces i moments aplicades sobre un eix.
Representació: Variables d'estat.
Tipus de control: Control *LQR*.

Exemple 3

Model del sistema: Utilitza la teoria de Lagrange basant-se en la diferència entre les energies cinètiques i potencial.
Representació: Variables d'estat.
Tipus de control: Control *PID*.

Conclusions

La manera més habitual d'extreure el model matemàtic és utilitzant la teoria de Lagrange, basant-se en la diferència d'energies cinètiques i potencials.
Tots els exemples coincideixen en el fet que la representació del sistema més comode i fàcil d'operar és la representació mitjanant variables d'estat.
Els dissenys de control que més avantatges aporten al model de pèndul invertit proposat en aquest treball són el control PID i el control LQR.

Capítol 3

Construcció i programació d'un model de pèndul invertit

En aquest capítol es descriuen: tots el components de *hardware* utilitzats en la construcció del model de pèndul invertit, la interfície de programació utilitzada i finalment la unió de totes les peces per obtenir el prototip de *segway*.

3.1 Parts *hardware* del prototip

3.1.1 Placa Arduino

Arduino és una plataforma d'electrònica oberta per a la creació de prototips basada en una placa, en un microcontrolador i una interfície gràfica de desenvolupament. El *hardware* de la placa Arduino consisteix en un microcontrolador ATmega328 de la casa Atmel AVR i ports d'entrada/sortida. D'altra banda, el *software* consisteix en un entorn de desenvolupament que implementa el lleguatge de programació Processing/Wiring. La placa Arduino és el sistema de procés emprat en el control del pèndul invertit d'aquest projecte.
Arduino es pot utilitzar per a desenvolupar objectes interactius autònoms o pot ser connectat a programari de l'ordinador (p.ex. Macromedia Flash, Processing, Max/MSP, Pure Data).

El model utilitzat per aquest projecte és l'Arduino Uno, detallat a continuació:

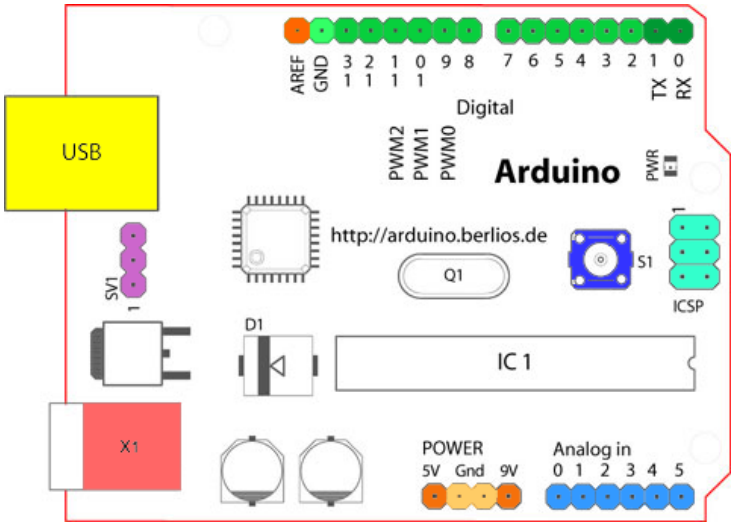


Figura 3.1: *Arduino board*

Entrades i Sortides:

Consta de 14 entrades digitals configurables d'entrada o sortida. Els pins 3,5,6,9,10,11 poden proporcionar una sortida **PWM**.
També compta amb 6 entrades analògiques que proporcionen una resolució de 10 bits.

Esquema de pins del microcontrolador Arduino:

- Pin de referència analògica (taronja).
- Senyal de terra digital (verd clar).
- Pins digitals del 2-13 (verd).
- Pins digitals 0-1, entrada/sortida del port sèrie TX/RX (blau).
- Botó de *reset* (negre).
- Entrada del circuit del programador sèrie (marro).
- Pins d'entrada analògica 0-5 (blau fosc).
- Pins d'alimentació i terra (taronja i taronja clar).

- Entrada de la font d'alimentació externa (gris X1).
- Port USB (vermell).

Es pot consultar més informació a la web oficial d'Arduino, veure ref.[1].

3.1.2 ATmega328P

El microcontrolador utilitzat és l'ATmega328P de la casa ATMEL.

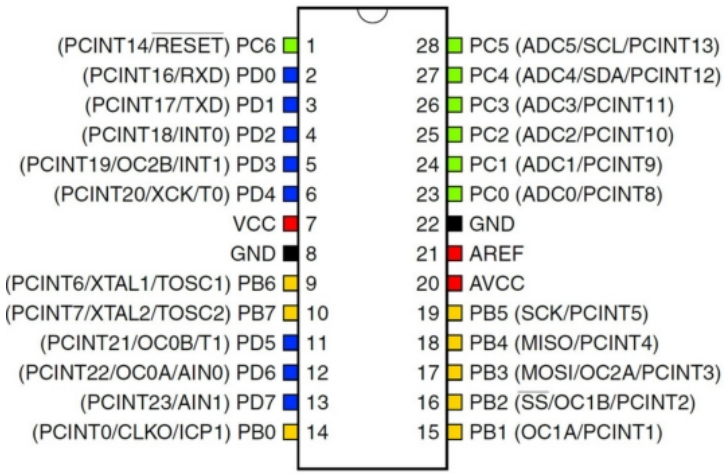


Figura 3.2: *Microcontrolador*

En la ref.[2] es pot trobar tota la informació tècnica més detallada, tal com les possibles configuracions dels pins per generar senyals analògics o digitals, informació referent als *timers*, possibles conflictes entre instruccions...

3.1.3 Xip l293D

Per tal de donar potència i canviar la configuració dels motors del pèndul invertit, és necessari el xip I293D.

El xip integrat l293d o, més conegut com a doble pont en H, inclou 4 circuits per dirigir càrregues de potència mitjanes (4.5v-36v), utilitzats en petits motors.

Cada xip l239D permet formar 2 ponts en H complets, és a dir, ens permet controlar 2 motors de forma independent. En aquest cas en concret els 2 motors reben el mateix control del microcontrolador, tenint així un aparell bidireccional de fre ràpid i amb control de velocitat.

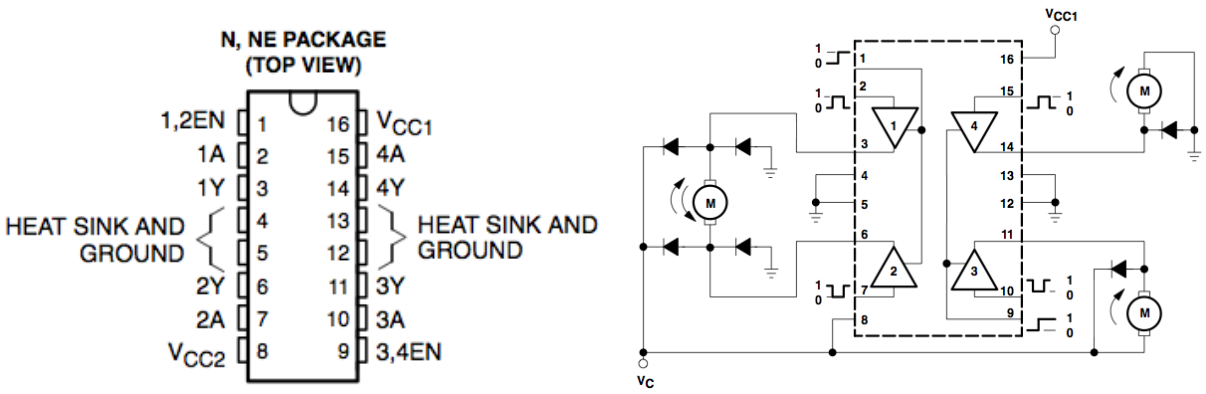


Figura 3.3: *L293d*

Esquema de pins del xip l293D:

- Pin 1 *enable* del xip. Està connectat a una sortida digital del microcontrolador. Sortida *low*(0) el xip està en *off*, i sortida *high*(1) el xip està en *on*.

- Pin 2 i 15 rep el senyal PWM encarregat del control de la velocitat, en l'apartat següent s'explica de forma més detallada.

- Pin 3 i 14 són sortides alternatives del xip que es connecten a dos motors respectivament. Aquests 2 pins s'han de connectar al mateix pol de cadascun dels motors, amb la finalitat que les dues rodes tinguin el mateix comportament.

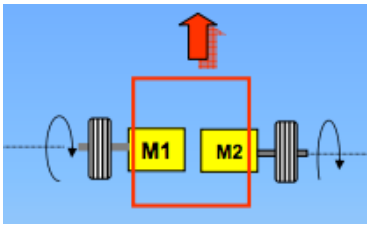


Figura 3.4: *Motors*

- Pin (4 i 5) i (12 i 13) són sortides a terra GND. És necessari tenir un pin de cada parella anterior (pont en H) connectat a terra.
- Pin 6 i 11 són dues sortides dels xips que van connectades als pols restants dels motors.
- Pin 7 i 10 són els que permet invertir el sentit de les rodes. Hi ha varis mètodes per fer-ho, en aquest cas en concret es troben connectats a una sortida digital binària *(low/high)* del microcontrolador, ja que quan prenen el valor *high* es converteixen en les entrades dominants i inverteixen el funcionament del motor.

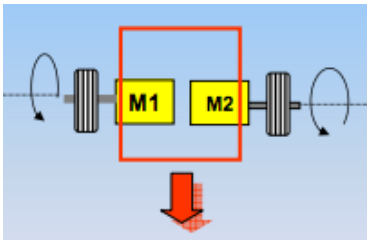


Figura 3.5: *Motors invertits*

- Pin 8 i 16 estan connectats al voltatge per tal d'alimentar el xip. S'han de connectar de tal manera que el voltatge del pin 16 (Vcc1)>= (Vcc2) del pin 8.

D'altra banda, la totalitat d'informació tècnica del xip I293D es pot observar en la bibliografia ref.[3], ref.[4].

Senyal PWM:

Tots els sistemes que processen informació binària per controlar un procés analògic necessiten convertidors ADC/DAC. D'altra banda, tots aquells que poden prescindir d'una elevada resolució, podem susbtituir els DAC per PWM, ref[2].

La modulació per amplada de pols o PWM(Pulse-With Modulation) és una tècnica en la qual es modifica el cicle de treball d'un senyal periòdic per controla la quantitat d'energia que s'envia a una càrrega.
Una unitat PWM permet assignar certa duració de temps en alt o en baix a una dada digital que es considera sortida d'una etapa processadora. Això s'aconsegueix connectant un comptador *timer* i un circuit comparador (OCR0B). ex: Quan el *timer* comença i fins que aquest arriba al valor del (OCR0B), la sortida PWM pren valor *high*(1), i des del valor (OCR0B) fins l'*overflow* del *timer* la sortida PWM pren valor *low*(0).
Quan es configura un *timer* per generar un senyal PWM, es divideix tot el recorregut del *timer* (16 bits) entre tots els valors que pot assolir el (OCR0B), que són 255. $(2^{16} - 1)/255$.

Així doncs, en el moment que la PWM està a 1 permet passar al motor tot el voltatge de la bateria a la qual està connectat. D'altra banda, quan està la PWM a 0 impedeix el pas del voltatge, mentre que en els estats intermitjos la PWM està intermitent i ens permet regular el voltatge que volem que entri al motor.

3.1.4 Motors DC.

El prototip és impulsat amb dos motors de corrent contínua. Aquests funcionen a un potencial de -5v a 5v.

Aquests pols del motor es connecten al xip I293D comentat anteriorment, un pol va al pin (3 o 14) i l'altre al pin (6 o 11).

Figura 3.6: Motor DC

3.1.5 Sensor MinIMU-9 Giroscopi, Accelerometre

Aquest dispositiu és el sensor emprat en el prototip per mesurar la inclinació del pèndul en temps real.

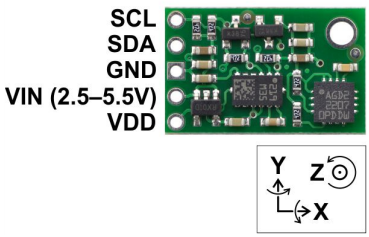


Figura 3.7: *Sensor giro/acc*

El Polulu MinIMU-9 és una unitat de mesura inercial IMU, la qual consta dels paquets L3G4200D amb 3 eixos per al giroscopi i LSM303DLM amb 3 eixos per l'acceleròmetre i el magnetòmetre. Val a dir que (durant aquest treball solament fou utilitzat l'acceleròmetre, no obstant això en la ref.[10] trobarem detalls tècnics sobre el sensor Min IMU 9, tals com la sensibilitat i la precisió, entre altres.

El MinIMU-9 admet un rang de valors de 16 bits 2^{16}, però al disposar de valors positius i negatius se li fa el complement a 2. Per tant, el rang de possibles valors queda reduït de -2^{16-1} a $2^{16-1} - 1$.

L3G4200D: és el sensor emprat per a mesurar l'orientació. Aquest es pot utilitzar per rastrejar amb molta precisió la rotació en un breu instant de temps.

LSM303DLM: és el sensor emprat per mesurar l'acceleració dels moviments. L'acceleròmetre pot ser útil per compensar la desviació del giroscopi amb el temps.

Aquest sensor es comunica amb el microprocessador emprant el protocol de comunicació I2C.

Esquema de pins:

- El pin *SDA* escriu i llegeix les dades.
- El pin *SGL* rep les senyals del rellotge del sistema.
- El pin *GND* va a terra.
- El pin *VDD* està connectat a la font d'alimentació.

Per a poder apreciar amb més detall el funcionament de l'acceleròmetre, en la figura 3.7 es mostren els diferents vectors de força implicats en el moviment del prototip del pèndul invertit.

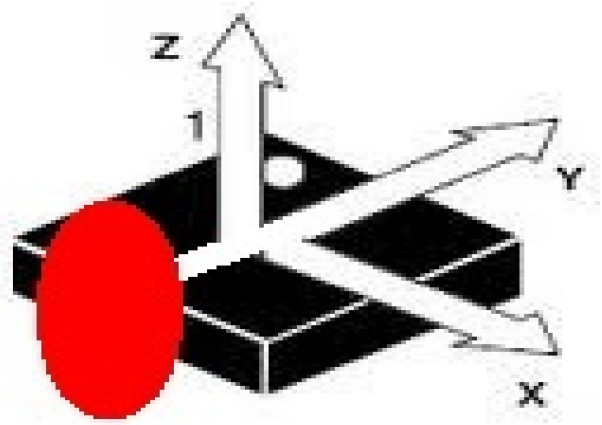


Figura 3.8: *Segway axis*

Tal i com es pot apreciar en la figura anterior, el prototip gira sobre l'eix Y . L'eix Z sempre tindrà un valor, ja que és la reacció aplicada per la gravetat, mentre que l'eix X ens mostra la inclinació del prototip a esquerra i dreta.

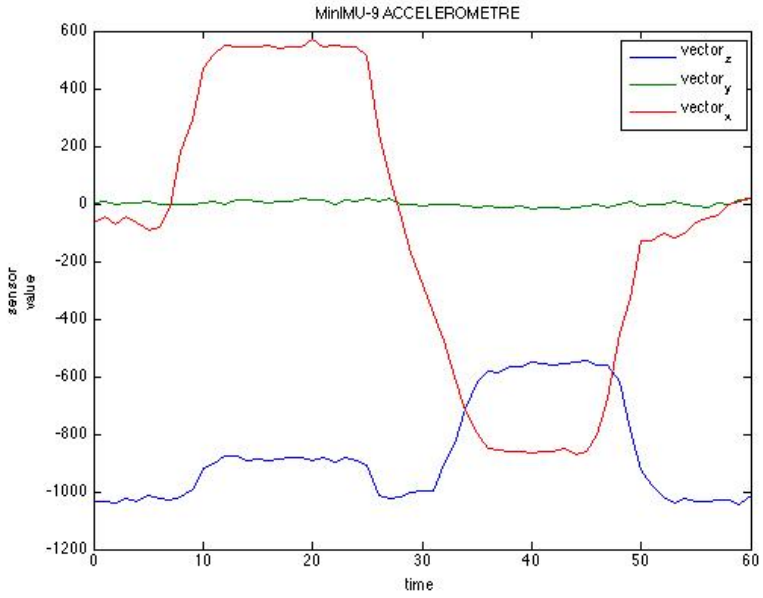


Figura 3.9: *MinIMU-9-accelerometre*

En la figura 3.8 s'observa una gràfica amb els valor adquirits pels 3 eixos quan el prototip està en estat inicial de repòs, inclinant-se a l'esquerra, inclinant-se a la dreta i finalment tornant a l'estat inicial de repòs.

Els pins SDA i SGL necessiten un protocol de comunicació I2C per comunicar-se amb el microcontrolador.

Tal i com s'ha explicat anteriorment, l'acceleròmetre et retorna les forces implicades en els tres eixos del sistema.

D'altra banda, per a extreure un model matemàtic més simple, utilitzarem graus d'inclinació de l'eix 'X_{acc}' [1] respecte l'eix ' Y_{acc}' [2], que anomenarem α.

El procediment utilitzat per a l'obtenció dels diferents graus d'inclinació (angles α) del nostre sistema és realitza mitjançant un procés manual, el qual consisteix en mantenir el carro en una posició estàtica i posteriorment anar movent la plataforma per tal d'obtenir diferents punts d'inclinació. Per cadascuna de les posicions en l'eix Y s'obté una mesura de l'acceleròmetre (figura 3.9). Finalment, per obtenir el valor de l'angle "α", solament és necessari resoldre la funció tan $\frac{y}{x}$ on es coneixen totes dues variables(distàncies).

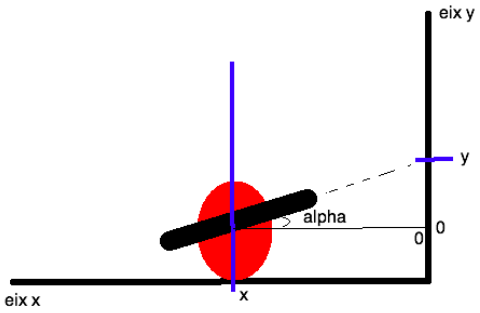


Figura 3.10: *Esquema de graus*

D'aquest procés s'obté la següent taula.

[1] X_{acc} (eix X acceleròmetre)
[2] Y_{acc} (eix Y acceleròmetre)

Dades de l'acceleròmetre	$\alpha = \tan \frac{y}{x}$
[532->546]	$\alpha = \tan \frac{4}{8} = 0.546 rads/seg = 31.30°$
[453->462]	$\alpha = \tan \frac{3.5}{8} = 0,467 rads/seg = 26.79°$
[308->322]	$\alpha = \tan \frac{3}{8} = 0.393 rads/seg = 22.55°$
[292->303]	$\alpha = \tan \frac{2.5}{8} = 0.323 rads/seg = 18.51°$
[221->248]	$\alpha = \tan \frac{2}{8} = 0,255 rads/seg = 14.63°$
[178->182]	$\alpha = \tan \frac{1.5}{8} = 0.189 rads/seg = 10.87°$
[111->135]	$\alpha = \tan \frac{1}{8} = 0.125 rads/seg = 7.19°$
[68 -> 84]	$\alpha = \tan \frac{0.5}{8} = 0.062 rads/seg = 3.58°$
[-4 -> 10]	$\alpha = \tan \frac{0}{8} = 0 rads/seg = 0°$
[-65 -> -82]	$\alpha = \tan \frac{-0.5}{8} = -0.062 rads/seg = -3.58°$
[-117-> -130]	$\alpha = \tan \frac{-1}{8} = -0.125 rads/seg = -7.19°$
[-177->-198]	$\alpha = \tan \frac{-1.5}{8} = -0.189 rads/seg = - - 10.87°$
[-246 -> -264]	$\alpha = \tan \frac{-2}{8} = -0.255 rads/seg = -14.63°$
[-297-> -315]	$\alpha = \tan \frac{-2.5}{8} = -0.323 rads/seg = -18.51°$
[-368->-382]	$\alpha = \tan \frac{-3}{8} = -0.393 rads/seg = -22.55°$

3.1.6 Sensor IR

Aquest aparell és un sensor de rajos infrarojos. S'utilitzà per comptar el número de radis que passaven per davant seu en un instant determinat de temps, amb la finalitat d'extreure el model matemàtic dels motors.

Figura 3.11: *Sensor IR i connexionat*

Tal com es pot apreciar a la figura 3.10, el sensor IR consta de tres conexions: una entrada GND que és el terra, una entrada V_{in} que és l'alimentació, i una sortida V_{out} connectada a una entrada digital del microcontrolador per tal de ser gestionat.

3.2 Arquitectura final del pèndul invertit

El sistema final es basa en la placa Arduino i els components que configuren el pèndul invertit. Com es mostra a la figura 3.12, en la placa s'hi connecten el sensor IR, el sensor Min-IMU 9, el xip l293d i la bateria. Al seu torn, els motors es troben connectats només al xip l293d. A continuació es mostra com es connecten tots els components descrits en l'apartat anterior.

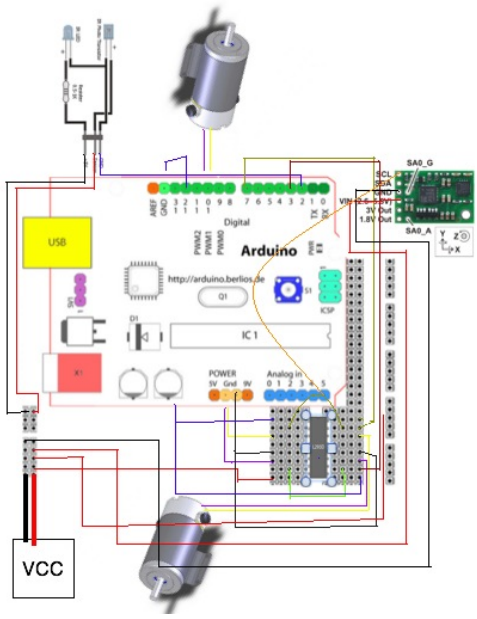


Figura 3.12: *Circuit del sistema*

La figura 3.13 mostra una imatge real del prototip, on s'identifiquen totes les parts que el composen.

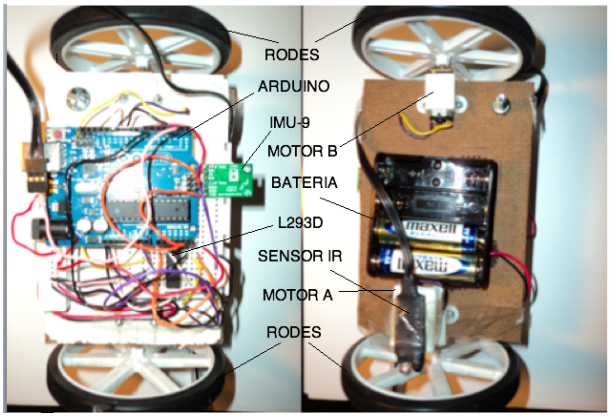


Figura 3.13: *Prototip final*

3.3 Disseny part *software* del prototip

En aquesta secció es descriu la GUI[1] utilitzada per desenvolupar el *software* i les parts base que ha de contenir un procés d'Arduino. El microncontrolador Arduino té un *software* propi per

[1]GUI (Graphical User Interface)

tal de programar les seves funcions amb una GUI molt senzilla i fàcil d'utilitzar.

3.3.1 GUI d'Arduino

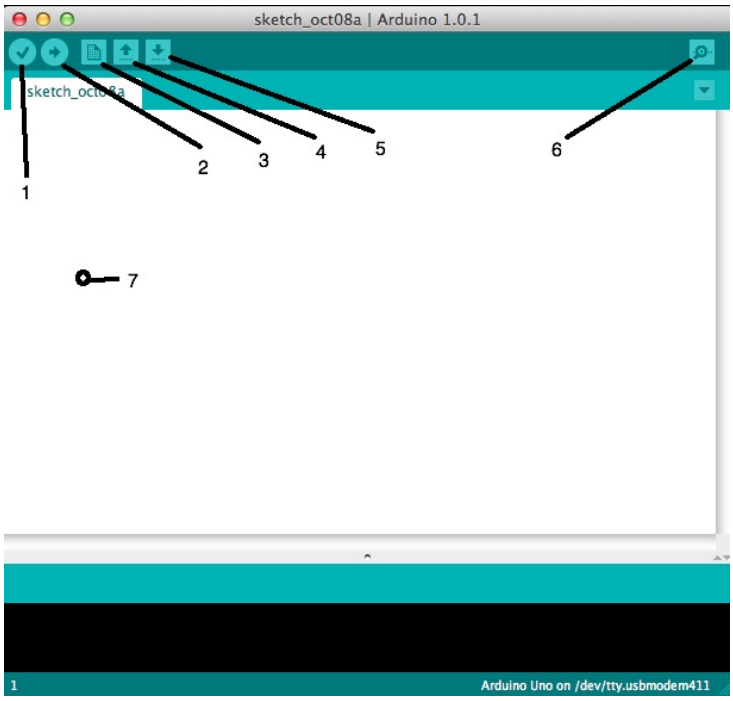


Figura 3.14: *Consola Arduino*

En la figura 3.14 s'observa la interfície del *software* Arduino utilitzat per a programar el microcontrolador Arduino UNO utilitzat en el prototip.
En la figura s'han especificat les principals icones que s'utilitzen en la interfície, en concret:

- 1. Serveix per verificar que no hi hagi cap error de compilació.
- 2. És el botó encarregat de descarregar el programa pel port sèrie al microcontrolador.
- 3. Botó de nova consola.
- 4. És el botó encarregat d'obrir les llibreries definides i exemples d'Arduino.
- 5. Serveix per guardar el codi.
- 6. Aquest botó obre una alra GUI que ens permet la comunicació bidirecional a través del port sèrie amb el microcontrolador.

- 7. Aquí és on l'usuari pot escriure el seu programa.

3.3.2 Estructura del programa basat en Arduino

Tots els processos programats en Arduino han d'incloure les següents parts:

- Al principi de tot el procés es posen les llibreries que s'inclouen, les definicions i les variables globals a tot el procés.
- Després va la inicialització (*set up*) del procés, que és on es configuren els ports(entrada, sortida), els *timers* i les interrupcions(internes o externes).
- Segueix el programa principal, el cos del codi. Aquesta funció s'anomena *void loop()* i es repeteix infinitament.
- Per últim, s'hi incorporaria la declaració de les funcions que fan d'interrupcions internes i externes.

Capítol 4

Model matemàtic del pèndul invertit

L'objectiu de la fase de modelat és trobar una expressió matemàtica que representi el comportament físic del sistema. Per a poder modelar el sistema, s'estudien els processos físics que hi tenen lloc i es dedueix una expressió matemàtica que descrigui el seu comportament (funció de transferència). Per a obtenir les equacions i calcular la funció de transferència es tria un algoritme ja implementat en la bibliografia ref.[6]. També cal destacar que el sistema de pèndul invertit es modela com un sistema lineal amb la finalitat de simplificar els càlculs, motiu pel qual aquest només serà vàlid per a oscil·lacions petites. A continuació, es descriu el procediment seguit per tal de realitzar el modelatge del sistema, format per un carro de dues rodes que constitueix un pèndul invertit.

4.1 Definició del problema:

El prototip consisteix en un carro amb només dues rodes, de manera que per si mateix el carro actua de pèndul invertit, ja que gira lliurement al voltant del pivot fix que és l'eix de les rodes. El prototip es mou com a conseqüència d'una força U provinent de dos motors de corrent contínua DC.

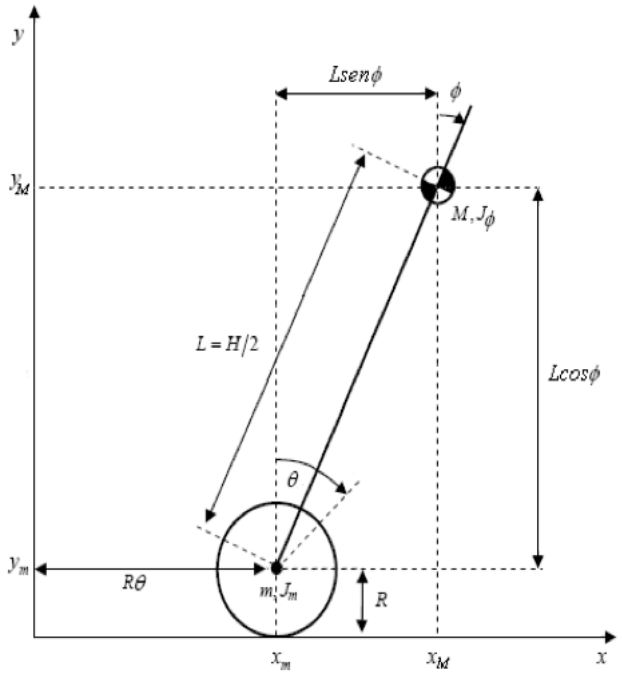


Figura 4.1: *Pèndul invertit*

Dades del sistema:

Abr.	Nom	Dades
m	Massa de les rodes.	0.250kg
M	Massa del pèndul.	0.02kg
b	Fricció del carro.	0,1N/m/sec
L	Distància del centre de masses al eix de les rodes.	0.035m
J_I	Inèrcia del pèndul.	m*($(l^2/12) + l)kg * m^2$
g	Gravetat.	9.8N
F	Força aplicada al carro.	N
X	Posició del carro.	x
Φ	Angle del pèndul des de la vertical.	ϕ

S'imposen uns requeriments mínims que hauria de complir davant una entrada esglaó:

- Temps d'establiment per 'x' i 'Φ' menor de 5 seg.
- Temps de pic per 'x' menor de 2 seg.

- Màxim sobrepic de 'Φ' menor a 20° (0.35rads).

4.2 Anàlisi de les forces i els sistemes d'equacions:

El pèndul invertit es pot entendre bàsicament com un cos rígid amb moviments limitats a dues dimensions.

Les equacions fonamentals de moviment en un pla d'un cos rígit són:

$$\sum_{i=1}^{\infty} F_i = m \cdot a_i \tag{4.1}$$

$$\sum_{j=1}^{\infty} F_j = m \cdot a_j \tag{4.2}$$

$$\sum_{g=1}^{\infty} F_g = I \cdot \alpha_g \tag{4.3}$$

Les equacions 4.1 i 4.2 són la suma total de les forces horitzontals i verticals respectivament. En l'equació 4.3, I és la suma total dels moments d'inèrcia. Sumant les forces en el diagrama del cos lliure del carro en la direcció horitzontal, s'obté la següent equació de moviment:

$$M\ddot{x} + b\dot{x} + N = F \tag{4.4}$$

Sumar les forces en direcció vertical no aporta cap infromació útil. D'altra banda, sumant les forces en direcció horitzontal es pot obtenir una equació per la variable N. On N és la força que exerceix el pèndul sobre el carro.

$$N = m\ddot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta \tag{4.5}$$

Substituint l'equació (4.5) en (4.4) s'obté la primera equació de moviment del sistema.

$$(M+m)\,\ddot{x} + b\dot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = F \tag{4.6}$$

Seguidament, és necessari sumar les forces perpendiculars al pèndul.

$$P\sin\theta + N\cos\theta - mg\sin\theta = ml\ddot{\theta} + m\ddot{x}\cos\theta \tag{4.7}$$

Per aïllar els termes P i N de l'equació anterior es sumen els moments al voltant del centre de gravetat del pèndul.

$$-Pl\sin\theta - Nl\cos\theta = J_i\ddot{\theta} \tag{4.8}$$

Finalment, combinant totes dues equacions, (4.7) i (4.8), obtenim la segona equació dinàmica.

$$\left(J_i + ml^2\right)\ddot{\theta} + mgl\sin\theta = -ml\ddot{x}\cos\theta \tag{4.9}$$

Les equacions 4.6 i 4.9 són les equacions de moviment del pèndul. Donat que el pèndul ha de garantir el seu funcionament per angles petits, podem linealitzar els sistema al voltant de $\theta = \pi$. Assumint que $\theta = \pi + \phi$ (ϕ representa un angle petit des de la vertical). Per tant, $\cos\theta = -1$, $\sin\theta = -\phi$ i $\left(\frac{d\theta}{dt}\right)^2 = 0$.
Un cop linealitzat el sistema d'equacions dinàmiques, obtenim les dues equacions següents:

$$(M+m)\,\ddot{x} + b\dot{x} - ml\ddot{\phi} = u \tag{4.10}$$

$$\left(J_i + ml^2\right)\ddot{\phi} - mgl\phi = ml\ddot{x} \tag{4.11}$$

4.3 Funció de transferència:

Qualsevol sistema físic es pot expresar com un conjunt de valors matemàtics a través dels quals es coneix el comportament d'aquests, davant una sèrie d'entrades concretes. La funció de transferència és un model matemàtic que, mitjançant un quocient, relaciona la resposta del sistema (sortida) amb un senyal d'entrada o exitació.
Per a obtenir la funció de transferència del sistema linealitzat analíticament, primer s'ha d'aplicar la transformada de Laplace a les equacions del sistema (suposant unes condicions inicials nules) ,que és la que ens dóna els valors de les variables en els domini 's', (X(s),U(s) i $\Phi(s)$).

$$\left(J_i + ml^2\right)\Phi(s)s^2 - mgl\Phi(s) = mlX(s)s^2 \tag{4.12}$$

$$\left(M+m\right)(s)s^2 + bX(s)s - ml\Phi(s)s^2 = U(s) \tag{4.13}$$

Aïllant X(s) en la primera equació tenim l'angle com a sortida del sistema.

$$X(s) = \left[\frac{(j_i + ml^2)}{ml} - \frac{g}{s^2}\right]\Phi(s) \tag{4.14}$$

I substituint l'equació (4.14) en (4.13):

$$(M + m\left[\frac{(j_i + ml^2)}{ml} - \frac{g}{s^2}\right]\Phi(s)s^2 + b\left[\frac{(j_i + ml^2)}{ml} - \frac{g}{s^2}\right]\Phi(s)s - ml\Phi(s)s^2 = U(s) \tag{4.15}$$

Agrupant els termes, resulta que la funció de transferència és:

$$\frac{\Phi(s)}{U(s)} = \frac{\frac{ml}{q}s^2}{s^4 + \frac{b(J_i+ml^2)}{q}s^3 - \frac{(M+m)mgl}{q}s^2 - \frac{bmgl}{q}s} \tag{4.16}$$

on

$$q = \left[(M+m)(J_i+ml^2) - (ml)^2\right] \tag{4.17}$$

Si cancel·lem el pol i el zero que tenen lloc en l'origen, ens queda la següent funció de transferència una mica més simplificada.

$$\frac{\Phi(s)}{U(s)} = \frac{\frac{ml}{q}s}{s^3 + \frac{b(J_i+ml^2)}{q}s^2 - \frac{(M+m)mgl}{q}s - \frac{bmgl}{q}} \tag{4.18}$$

4.4 Variables d'estat

Expressant les equacions del sistema en forma de variables d'estat amb el mètode vist en: ref.[6].

$$\begin{bmatrix} \dot{X} \\ \ddot{X} \\ \dot{\Phi} \\ \ddot{\Phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(J_i+ml^2)b}{J_i(M+m)+Mml^2} & \frac{m^2gl^2}{J_i(M+m)+Mml^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-mlb}{J_i(M+m)+Mml^2} & \frac{mgl(M+m)}{J_i(M+m)+Mml^2} & 0 \end{bmatrix} \begin{bmatrix} X \\ \dot{X} \\ \Phi \\ \dot{\Phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{J_i+ml^2}{J_i(M+m)+Mml^2} \\ 0 \\ \frac{ml}{J_i(M+m)+Mml^2} \end{bmatrix} u \tag{4.19}$$

$$Y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ \dot{X} \\ \Phi \\ \dot{\Phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u \tag{4.20}$$

Seguint la següent estructura:

$$\dot{X} = A + B \cdot U$$

$$Y = C \cdot X + D \cdot U$$

És important destacar que, tant la posició del carro (X en metres) com l'angle de desviació del pèndul (ϕ en radiants) formen part de la sortida. Per tant, han d'estar representats en la matriu C.

4.5 Resposta en llaç obert del sistema

Un cop es disposa del model del sistema, s'ha d'analitzar el seu comportament en llaç obert per comprovar l'estabilitat del sistema i decidir si és necessari un control en llaç tancat.

4.5.1 Per funció de transferència

En aquesta secció, s'analitza el sistema modelat amb funció de transferència H(s). En el cas de la H(s) només es tracta d'un sistema amb una única sortida.

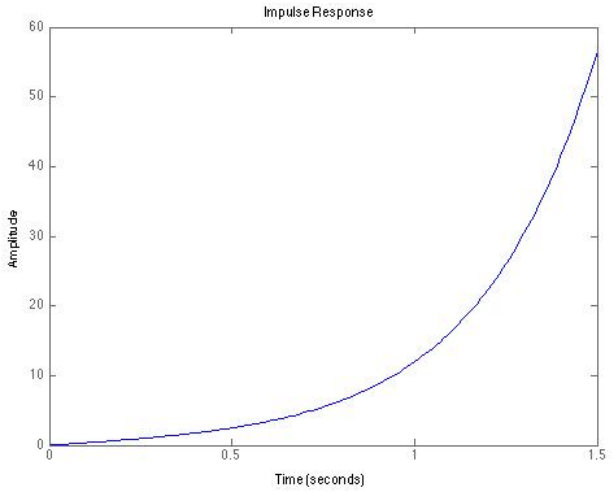


Figura 4.2: *H(s) en llaç obert*

La figura 4.2 mostra com el sistema és inestable, ja que els valors de la sortida tendeixen a infinit i no aconsegueixen mai un valor constant. Donat que el sistema no es estable en llaç obert és necessari el disseny d'un controlador.

4.5.2 Per variables d'estat

En aquesta secció es comprova si el sistema per variables d'estats (VVEE) és estable en llaç obert. El sistema VVEE és un sistema multisortida, per tant es pretén controlar l'angle del pèndul i la posició del carro, mitjançant l'aplicació d'aquest.

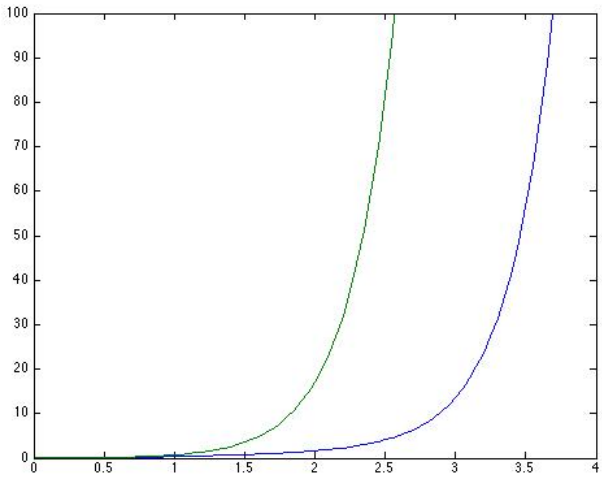


Figura 4.3: *VVEE amb llaç obert*

En la gràfica anterior s'observa la sortida del sistema, on la línia blava és la posició del carro i la verda és l'angle del pèndul.
Com en la figura (4.2), s'haurà d'incloure un mòdul de control per tal que el sistema s'acoti el màxim possible als requeriments mínims plantejats anteriorment.

4.6 Model del motor DC

Per a extreure el model matemàtic del motor, s'han utilitzat dos *scripts* diferents que es poden trobar als apèndixs A i B d'aquest treball.
El primer *script* plantejat és un *script* arduino amb les següents característiques:

- Envia seqüencialment els següents senyals *pwm* als motors $[0, 50, 0, 100, 0, 150, 0, 200, 0, 250, 0]$.
- Una interrupció externa intervé cada cop que un radi passa per davant del sensor IR.
- Una interrupció interna imprimeix cada 0.1 seg. en un fitxer, que serà el que a posteriorment se li introduirà a l'script de Matlab.

El segon *script* és un fitxer Matlab, el qual realitza les següents operacions:

- Amb el fitxer rebut es calculen uns *plots* amb: la potència donada, la velocitat obtinguda i una relació amb la velocitat i l'estat dels motors.
- Extreu un model del sistema amb continu, i les seves arrels.
- Extreu un model del sistema amb discret, i les seves arrels.

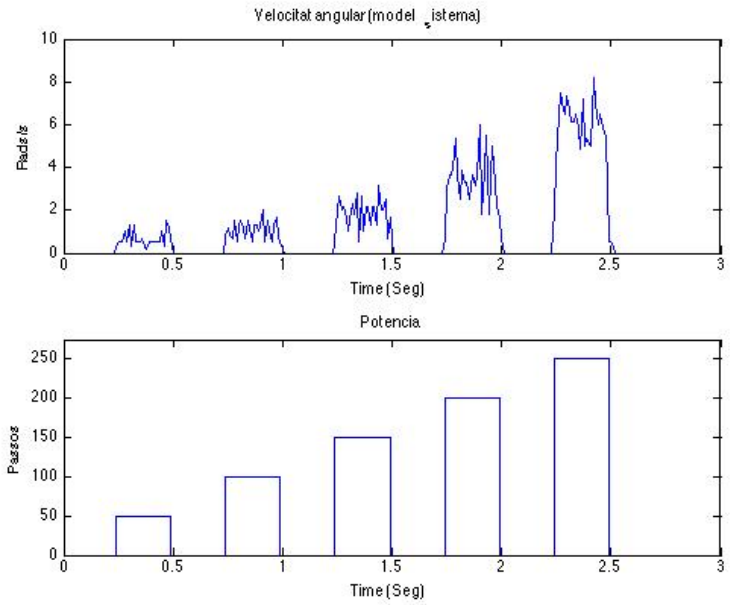


Figura 4.4: *Potència d'entrada* vs *Velocitat de sortida*

En la figura (4.4) s'observa com la 2a gràfica és el conjunt de potències distribuides seqüencialment al motor, mentre que la 1a gràfica és la velocitat de resposta que correspont a cada potència. Finalment en la figura (4.5) es pot apreciar com es relaciona la velocitat de sortida amb el retard del motor a l'hora d'assolir la potència subministrada.

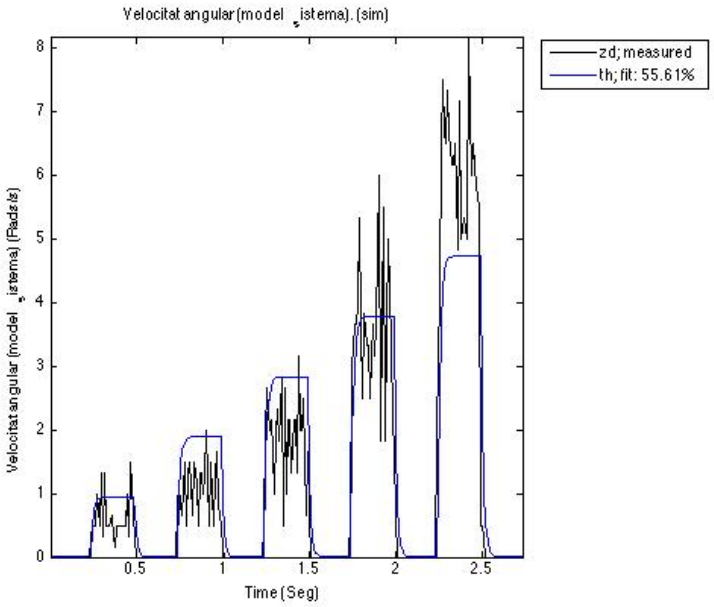


Figura 4.5: *Resposta del motor*

Model del motor amb continu:

$$H(s)_{motor} = \frac{1.4063}{s + 74.2386}$$

Model del motor amb discret $t_s = 0.01segs$:

$$H(z)_{motor} = \frac{0.0099268}{z - 0.475986}$$

Capítol 5

Disseny del control

Com s'ha comprovat en el capítol anterior, la dinàmica del sistema per si sola és incapaç de mantenir el pèndul estable, independentment de si es modela amb el mètode de funció de transferència o amb el de variables d'estat.
En aquest capítol, s'introduirà el concepte de regulador o controlador per tal d'evitar que el sistema es desestabilitzi. Existeixen diversos mètodes per dissenyar un regulador. A continuació detallem dos tipus en concret, un regulador PID per al cas del model amb funció de transferència i un regulador LQR per al disseny amb variables d'estat. Tot i que tots dos són molt interessants a l'hora de realitzar el disseny ,el que finalment s'implementarà en el prototip és el regulador LQR, ja que proporciona una estabilització de la posició del carro. Per contra, el PID que no ho aconsegueix, tal i com es mostra en els apartats següents.

5.1 Controlador PID

Els controladors PID formen part dels controladors clàssics en la història de l'enginyeria de control, degut a la seva robustesa i simplicitat d'implementació.
En aquesta secció no s'expliquen els fonaments teòrics d'aquest controlador, ja que són detallats en el tercer exemple del capítol dos. D'altra banda s'explica el seu funcionament mitjançant l'exemple representat en el següent punt.

5.1.1 Disseny d'un controlador PID

L'objectiu d'aquest apartat és veure com es calcula manualment un controlador PID amb un exemple senzill, ja que amb el cas implementat en el prototip s'utilitzarà el control programat en Matlab i no es pot apreciar la totalitat del seu funcionament.
Primer s'extreu el llaç tancat del diagrama de blocs.

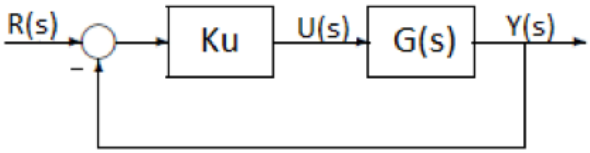


Figura 5.1: *Diagrama de blocs del sistema*

$$ll_{tancat} = \frac{K_u G(z)}{1 + K_u G(z)} \tag{5.1}$$

La forma més comuna de dissenyar un PID és mitjançant l'aproximació de Ziegler-Nichols. Bàsicament consiteix en trobar els valors K_c, T_i i T_d de la taula de Ziegler-Nichols amb el valor de K_u prèviament calculat. Un cop calculats aquests valors es substitueixen a l'equació bàsica del regulador i ja tenim un controlador PID per al nostre sistema.
Equació bàsica del regulador PID pel mètode d'aproximació de Ziegler-Nichols:

$$C(z) = \frac{a_0 z^2 + a_1 z + a_2}{z(z-1)} \tag{5.2}$$

$$a_0 = K_c\left(1 + \frac{T_d}{T}\right); a_1 = K_c\left(-1 + \frac{T}{T_i} - \frac{2T_d}{T}\right); a_2 = K_c\frac{T_d}{T}; \tag{5.3}$$

Els paràmeres d'aproximació per a un controlador PID són els següents:

$$K_c = 0.6K_u; T_i = \frac{T_u}{2}; T_d = \frac{T_u}{8};$$

on,

$$T_u = \frac{2\pi}{W_u}; W_u = \frac{1}{T}\tan^{-1}\left(\frac{\beta}{\alpha}\right) \tag{5.4}$$

Exemple:

Per al següent exemple la funció de transferència serà:

$$G(z) = \frac{1}{z^2 + 2z + 3} \tag{5.5}$$

En primer lloc s'extreu l'equació característica, que no és res més que el denominador de l'expresió en llaç tancat del diagrama de blocs. Per tant, l'equació característica corresponent a aquest exemple és:

$$1 + K_u \frac{1}{z^2 + 2z + 3} \tag{5.6}$$

A continuació, es resol l'equació de 2n grau mantenint la variable k_u i forçant a que el mòdul de z sigui igual a 1 ($|z| = 1$).

$$z = \frac{-2 \pm \sqrt{2^2 - 4(3 + k)}}{2} \tag{5.7}$$

$$|z| = 1 \quad = \quad \sqrt{\left(\frac{-2}{2}\right)^2 + \left(\frac{\sqrt{-8 - 4k}}{2}\right)^2} \tag{5.8}$$

$$1 \quad = \quad 1 + \frac{-8 - 4k}{4} \tag{5.9}$$

Finalment, obtenim el valor de $k = -2$. A continuació, pel valor de k obtingut extraiem les arrels del sistema.

$$z = \frac{-2 \pm \sqrt{4 - 12 - 4 \cdot 2}}{2}; \frac{-2 \pm \sqrt{0}}{2}; -1 \pm 0j \tag{5.10}$$

Ara, ja només falta calcular T_u per poder fer l'aproximació de Ziegler-Nichols. Suposem que el valor del període és de 0.1.

$$W_u = \frac{1}{0.1}\left(\tan^{-1}\left(\frac{0}{-1}\right) + \pi\right) = 31.42 \tag{5.11}$$

$$T_u = \frac{2\pi}{31.42} = 0.199 \tag{5.12}$$

Ja només queda substituir els valors T_u i W_u a la taula de Ziegler-Nichols per a un controlador PID, i obtenim:

$K_c = 0.6 \cdot K_u = 0.6 \cdot -2 = -1.2$

$T_i = \frac{T_u}{2} = \frac{0.199}{2} = 0.095$

$T_d = \frac{T_u}{8} = \frac{0.199}{8} = 0.0248$

El controlador per al sistema seria:

$$C(z) = \frac{-1.4976z^2 - 0.5320z - 0.2976}{z(z-1)} \tag{5.13}$$

5.1.2 PID aplicat en un pèndul invertit

En aquest apartat es mostra com s'estabilitza el prototip de pèndul invertit. Per fer-ho s'utilitza Matlab com a eina matemàtica per realitzar tots els càlculs. Tot el codi Matlab per al disseny del controlador *PID*, per al sistema amb funció de transfèrencia, es pot trobar a l'apèndix C(PID per a TF).

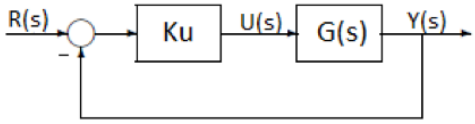


Figura 5.2: *TF Diagrama de blocs del sistema.*

Primer s'analitzarà l'estabilitat del sistema tenint en compte la inclinació(rads). Per aconseguir això, primer cal presentar la funció de transferència del prototip. Partint de l'equació(4.18) presentada en el capítol 4 i les dades que descriuen el prototip de pèndul invertit en aquest mateix capítol, s'obté la següent equació de tranferència:

$$pend(s) = \frac{5.18s}{s^4 + 74.62s^3 + 18.64s^2 - 727.2s - 268} \tag{5.14}$$

Per poder analitzar l'estabilitat del sistema *pend(s)*, és necessari conèixer on es troben totes les arrels del sistema. Per tal d'obtenir aquesta informació utilizem la comanda Matlab *rlocus(G(s))*, que retorna el lloc de les arrels amb les asimptotes que ens mostren la tendència d'aquestes amb el pas del temps.

Tal i com s'observa a la figura 5.3, aquest sistema és inestable, ja que pràcticament des de l'instant 0 totes les arrels, a exepció d'una, estan en el semipla dret. L'objectiu d'afegir un controlador en el sistema plantejat és aconseguir traslladar la tendència d'aquestes arrels al semipla esquerre per aconseguir l'estabilitat del sistema.

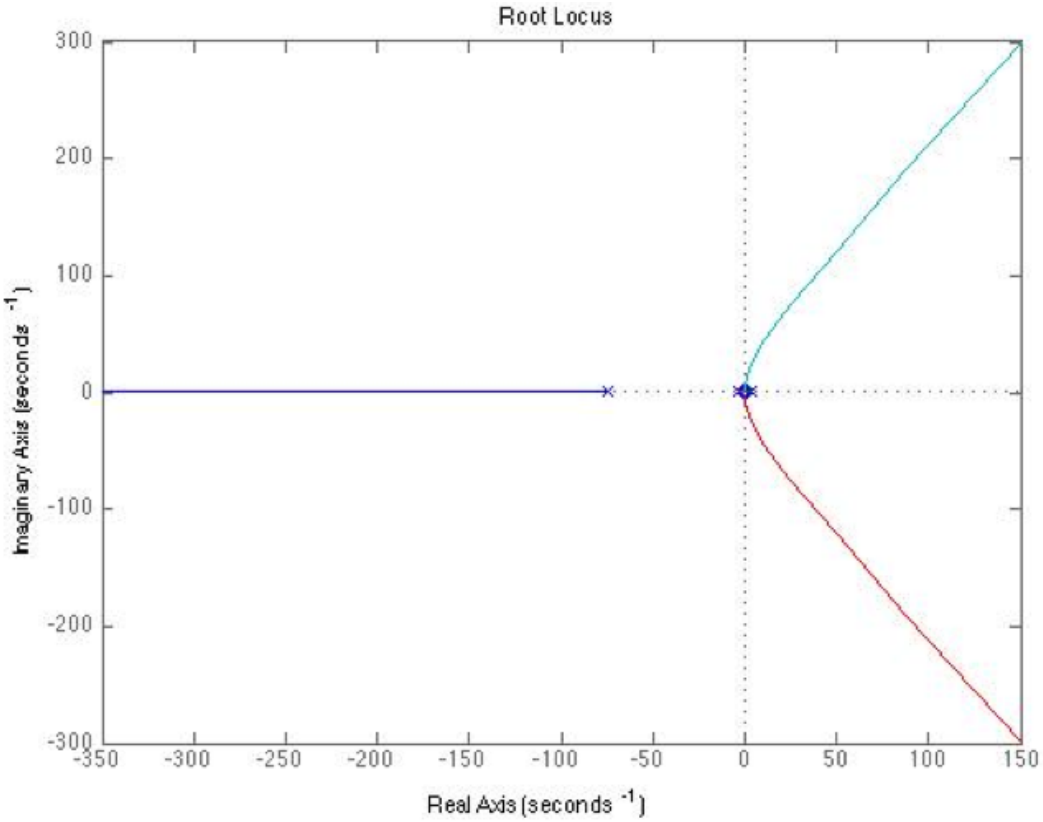


Figura 5.3: *TF Lloc de les arrels del sistema*

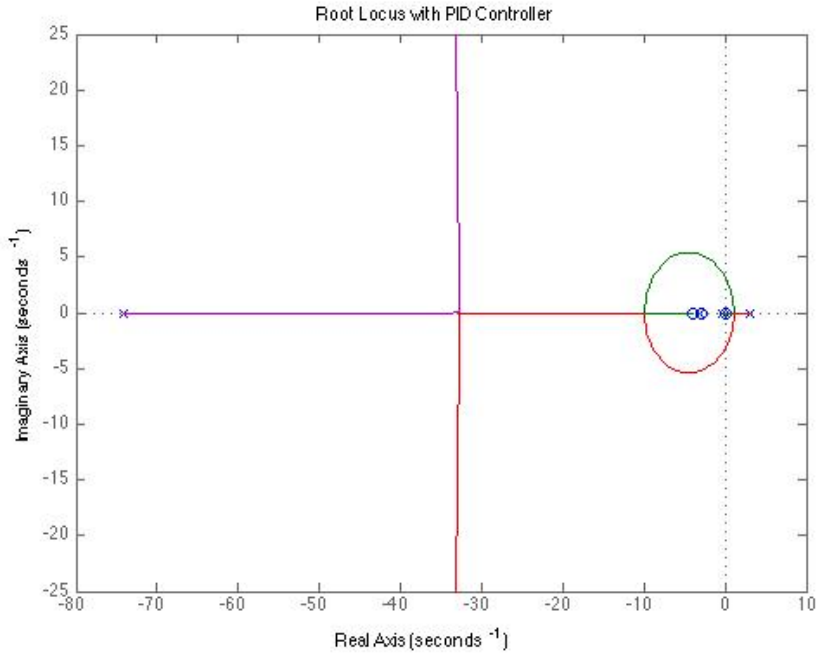


Figura 5.4: *TF Lloc de les arrels del sistema+PID*

Un cop introduït el control al sistema, en la figura 5.4 s'observa com les arrels anteriors en el semipla dret ja es troben situades al semipla esquerre, fet que ocasiona l'estabilització del sistema. A continuació es mostren les instruccions Matlab més rellevants que fan possible l'obtenció d'un controlador amb les perspectives del sistema.

- *C = zpk(z,p,k)*

  Zpk retorna un controlador C amb les especificacions que se li passen com a paràmetres, z zeros, p polos i k guany.

- *[k,poles] = rlocfind(C*pend)*

  Retorna el guany que ha de tenir el sistema, per a que aquest treballi al punt indicat en la gràfica. També retorna els pols del sistema.

- *K = 100;*

  Aquest és el guany corresponent al punt escollit en el *root locus*.

- *T = feedback(pend*K*C,1);*

El *feedback* treu la funció corresponent al llaç tancat del sistema.

- *impulse(T)*

  Impulse retorna una gràfica del comportament del sistema en llaç tancat.

El controlador escollit és:

$$C = \frac{(s+3)(s+4)}{s} \tag{5.15}$$

I com ja s'ha explicat anteriorment, el guany serà de K=100, que és el guany requerit per tal que el sistema treballi en la regió escollida.
Per tant, el llaç tancat del sistema final és: $lltancat = feedback(pend * K * C, 1)$

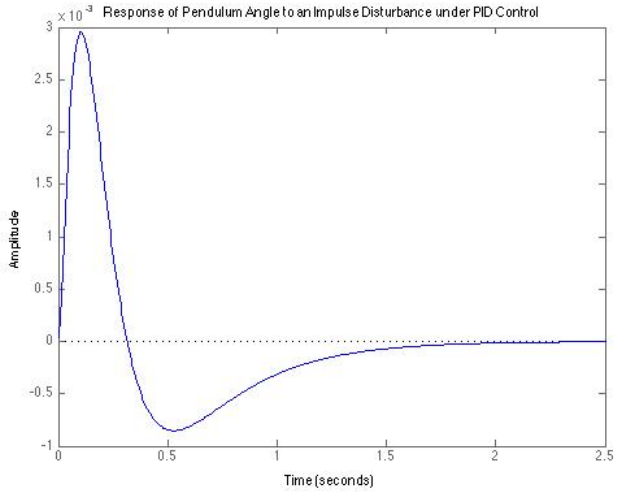


Figura 5.5: *TF Resposta de la inclinació a una entrada impuls.*

Així doncs, tal i com es mostra en la figura 5.5, el sistema és estable passats 2 segons. Cal destacar l'elevada pertorbació que s'observa al principi, degut a la restricció que imposem en el sistema per tal que sigui estable a partir dels 2 segons de control. Si posessim els zeros del controlador molt més a l'esquerra, el temps d'estabilització del sistema seria molt més elevat però el sobrepic seria inferior.
Un cop estabilitzada la inclinació del sistema anem a veure que passa amb la posició del carro. La funció de transferència que descriu la posició del carro és la següent:

$$ft_{carro} = \frac{3.823s^2 - 36.1}{s^3 + 0.3823s^2 - 9.746s - 3.61} \tag{5.16}$$

A continuació es simula el sistema amb la funció de transferència en llaç tancat, que inclou la posició del carro. Per a la simulació d'aquest s'utilitza el mateix control C, planta *pend* i instruccions Matlab emprades en la simulació anterior.

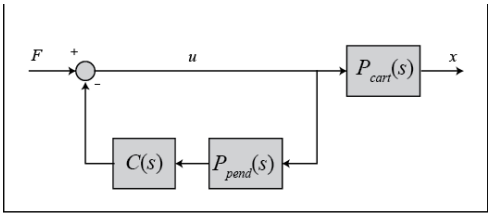


Figura 5.6: *TF Diagrama de blocs del sistema+posició.*

El llaç tancat d'aquest nou diagrama de blocs és: $ll_{tancat} = \frac{ft_{carro}}{(1+pend*C)}$

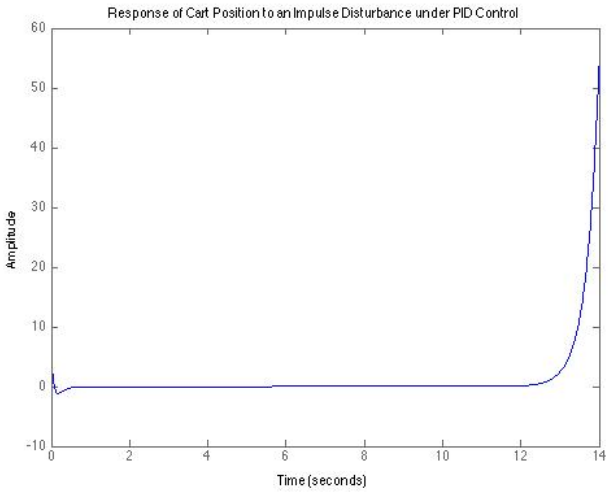
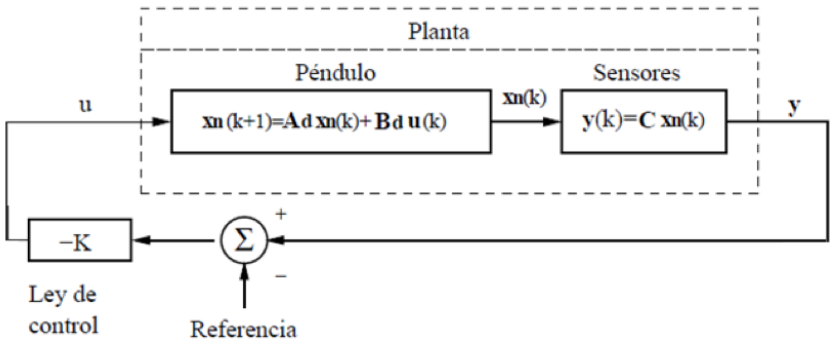


Figura 5.7: *TF Resposta de la posició del carro a una entrada impuls.*

En la figura anterior, la posició del carro s'estabilitza en poc més de 0,4 segons, però després de 12 segons es desestabilitza. Aquest fet, juntament amb el gran sobrepic de quasi 3 radiants —vist en la figura 5.5—, fan que no sigui factible aplicar aquest control sobre el nostre sistema físic real. En conseqüència, el control PID queda descartat per al nostre prototip de pèndul invertit.

5.2 Controlador LQR

Seguidament, es proposa dissenyar un controlador LQR(*Linear Quadratic Regulator*) per tal que compleixi amb les especificacions de controlabilitat definides per al sistema. A causa de la representació del sistema amb variables d'estat, el disseny del controlador LQR resultarà ser més fàcil d'implementar. En el sistema real, no disposem de totes les variables necessàries per a realitzar el control, per tant serà necessari utilitzar un observador per estimar part del valor d'aquestes variables. Per saber si és possible o factible el disseny d'aquest observador, primer que tot s'ha d'extreure l'observabilitat del sistema, que és el que ens dirà si és possible estimar tots aquests valors. Un cop comprovada l'observabilitat, es dissenyarà l'observador d'ordre reduït.
La següent figura mostra el diagrama de blocs del sistema implementant un controlador LQR:



Al tema 2 ja s'ha fet una introducció teòrica als controladors lineals quadràtics, per tant, tot seguit veurem el disseny de control LQR proposat per al nostre cas concret de pèndul invertit.

5.2.1 Disseny d'un LQR en espai d'estats

Un cop analitzada la base teòrica, ja es pot començar amb el disseny del controlador LQR per a un model en espai d'estats. A continuació veurem com es desenvolupa el procés d'obtenció de l'algorisme de control LQR amb Matlab.
Es pot trobar el codi complet del disseny del controlador LQR per a un model del sistema per variables d'estat en l'apèndix F.

Discretització del sistema

Com que s'implementa el control en un microprocessador, el qual opera amb dades discretes juntament amb els sensors que envien informació d'una forma periòdica, serà necessari discretit-

zar el sistema obtingut.

Per discretitzar el nostre sistema representat per variables d'estat en forma matricial, Matlab disposa de la instrucció *c2dm*. Aquesta funció retorna 4 matrius, que són les matrius que representen el sistema amb variables d'estat discretitzades. Per a que *c2dm* funcioni necessita 6 arguments: les 4 matrius que representen el sistema en variables d'estat continu(A,B,C,D), un període de mostreig (T_s $sec/mostra$) i el mètode de mostreig, l'escollit és el zero-order-hold ('zoh').

Cal destacar que, segons el teorema de mostreig de Nyquist-Shanon, en què es pot recosntruir una senyal analògica a partir d'una senyal digital sense perdre informació, si i només si, la freqüència de mostreig és igual o més gran a dues vegades l'ample de banda(AB) $f_m \geq 2 \cdot AB$. Suposant que l'AB en llaç tancat és de 1rad/seg per al pèndul, introduirem un període de mostreig de 0.01 seg/mostra.

El resusltat de *c2dm* són les 4 matrius (F,G,H,J) que representen el sistema en un temps discret t_s.

$Ts = 1/100$; *% període de muestreig*

$[F, G, H, J] = c2dm(A, B, C, D, Ts,' zoh')$;

A partir d'ara treballarem amb la següent representació del sistema amb variables d'estat, ja que, al discretitzar-la, s'han modificat certs paràmetres físics respecte al disseny original:

$$\begin{bmatrix} \dot{X}(k) \\ \ddot{X}(k) \\ \dot{\Phi}(k) \\ \ddot{\Phi}(k) \end{bmatrix} = \begin{bmatrix} 1 & 0.01 & 0 & 0 \\ 0 & 0.9962 & 0.0032 & 0 \\ 0 & 0 & 1 & 0.01 \\ 0 & -0.0037 & 0.0975 & 1 \end{bmatrix} \begin{bmatrix} X(k-1) \\ \dot{X}(k-1) \\ \Phi(k-1) \\ \dot{\Phi}(k-1) \end{bmatrix} + \begin{bmatrix} 0.0002 \\ 0.0382 \\ 0.0002 \\ 0.0368 \end{bmatrix} u(k-1) \tag{5.17}$$

$$Y(k-1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X(k-1) \\ \dot{X}(k-1) \\ \Phi(k-1) \\ \dot{\Phi}(k-1) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u(k-1) \tag{5.18}$$

Representació en forma matricial:

$$x(k) = F \cdot x(k-1) + G \cdot u(k-1)$$
$$y(k) = H \cdot x(k-1) + J \cdot u(k-1)$$

Controlabilitat i observabilitat

Un sistema és controlable en un instant de temps t_0 si es pot portar des de qualsevol estat inicial $x(t_0)$ a qualsevol altre estat mitjançant un vector de control, en un interval de temps finit. Un sistema és observable en un instant de temps t_0 si amb el sistema en un estat $x(t_0)$ és possible determinar aquests estat a partir de l'observació de la sortida, amb un instant finit de temps. El concepte d'obsevabilitat és útil per resoldre el problema de reconstrucció de variables d'estat no mesurables en aquelles que sí ho són.
Per tal que un sistema sigui completament controlable i observable, les matrius corresponents de controlabilitat (C_T) i observabilitat(O_T) han de tenir rang $\boldsymbol{n}$ ref[8].

$$C_T = \begin{bmatrix} G & FG & F^2G & ... & F^{n-1}G \end{bmatrix}; O_T = \begin{bmatrix} H \\ HF \\ HF^2 \\ ... \\ HF^{n-1} \end{bmatrix};$$

Llavors, com el resultat de les matrius de controlabilitat i l'observabilitat corresponents són de 4x4, el seu rang ha de ser de 4. A continuació s'analitza el sistema per comprovar la controlabilitat i la observabilitat amb intruccions Matlab.

$co = ctrb(F, G);$

$Controllability = rank(co)$

$ob = obsv(F, H);$

$Observability = rank(ob)$

Aquesta part de codi crea les matrius de controlabilitat i observabilitat, i després n'extreu els rangs corresponents . Així doncs en el nostre cas en concret serà:
Controllability=4; Observability=4;
Com que el rang de les matrius C_T i O_T és d'ordre 4, queda demostrat que el sistema és completament controlable i observable.

Disseny d'un controlador LQR

A continuació, en la figura 5.9 es mostra un sistema de realimentació de variables d'estat complet, on R representa el senyal de referència del carro, u és el senyal de control que entra al

sistema definit per les matrius *[F,G,H,J]*, K és la matriu de control, x és el vector de variables d'estat i y és el vector que conté la sortida del sistema.

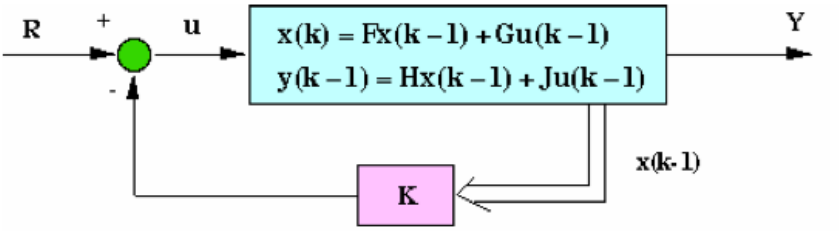


Figura 5.8: *LQR diagrama de blocs del sistema*

Per a l'obtenció de la matriu K s'utilitza la instrucció Matlab *dlqr*. Aquesta instrucció retorna un controlador òptim, resultat de fer un balanç entre els errors del sistema i l'esforç de control. La instrucció *dlqr* requereix de 4 paràmetres, 2 dels quals ja els tenim (les matrius del sistema H iG); la matriu d'índex d'actuació R i la matriu de cost d'estats Q' són les dues restants. Per a l'obtenció de la matriu Q serà necessari definir els factors de pes. Els factors de pes són el pes que serà atorgat a cadascuna de les variables d'estat dintre de la matriu Q. Aquestes estan representades en la diagonal de la matriu, on cada posició és una variable d'estat. En aquest treball, només es defineixen les que poden ser controlades, que són la posició del carro *Q[1,1]*, que anomenarem (x) i l'inclinació del pèndul *Q[3,3]* que anomenarem (y).
Com ja s'ha explicat anteriorment, el cas més simple és considerar R=1 i treure els paràmetres x i y de Q per prova i error ref.[8].
Tot seguit es mostra el codi per a l'obtenció i la simulació del controlador.

$figure(1)$

$T = 0 : 0.01 : 5;$

$U = 0.2 * ones(size(T));$

$x = 5; \%factordeponderacipERlaposicidelcarro$

$y = 0.1; \%factordeponderacipaerl'angledelp?ndul$

$Q = [x000;$

$0000;$

$00y0;$

$0000];$

$R = 1;$

$K = dlqr(F, G, Q, R)$

$[Y, X] = dlsim(F - G * K, G, H, J, U);$

$stairs(T, Y)$

$legend('Cart(x)', 'Pendulum(phi)')$

Els valors x i y s'han anat modificant fins arrivar a $x = 5$ i $y = 0.1$

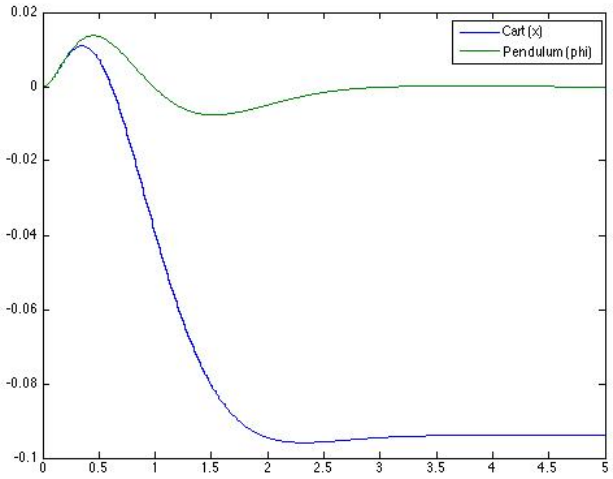


Figura 5.9: *Sistema amb control lqr*

Tal i com s'observa a la Figura 5.10, el sistema compleix amb els requeriments temporals descrits en l'apartat 4.6, i per tant es dóna el control com a vàlid. La corba superior representa l'angle d'inclinació del pèndul amb radiants i la corba inferior representa la posició del carro davant una entrada esglaó de 0.2m. D'altra banda, si analitzem la posició del carro, s'observa que no és gens satisfactòria. De fet, el carro s'ha mogut en la direcció contrària. Així doncs, per tal de corregir aquesta desviació, ens veiem obligats a introduir un senyal de referència, tal i com es comenta en el següent apartat.

Entrada de referència

Degut a que aquest sistema de control no compara l'entrada amb la sortida, sinó que ho fa amb el resultat de multiplicar totes les variables d'estat per la matriu K (K*x), és improvable que la sortida sigui igual a l'entrada. Per a escalar la sortida, per tal que sigui igual o molt similar a l'entrada, és necessari introduir un valor de referència.

Després de realitzar un conjunt de proves obtenim el valor de $N_{bar} = -2.12$.
En el diagrama de blocs següent, es pot apreciar on es posa l'entrada de referència, per tal que la sortida sigui el valor desitjat.

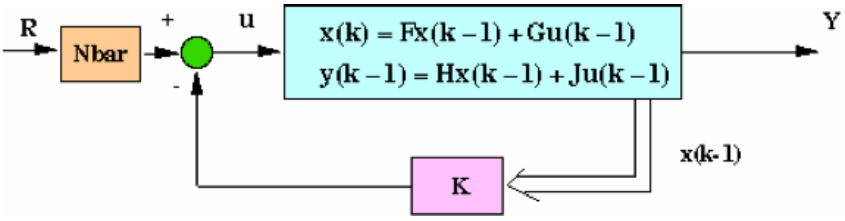


Figura 5.10: *Diagrama de blocs+ referencia*

Seguidament, a la Figura 5.12 s'oberva que hem aconseguit que la posició del carro sigui la desitjada.

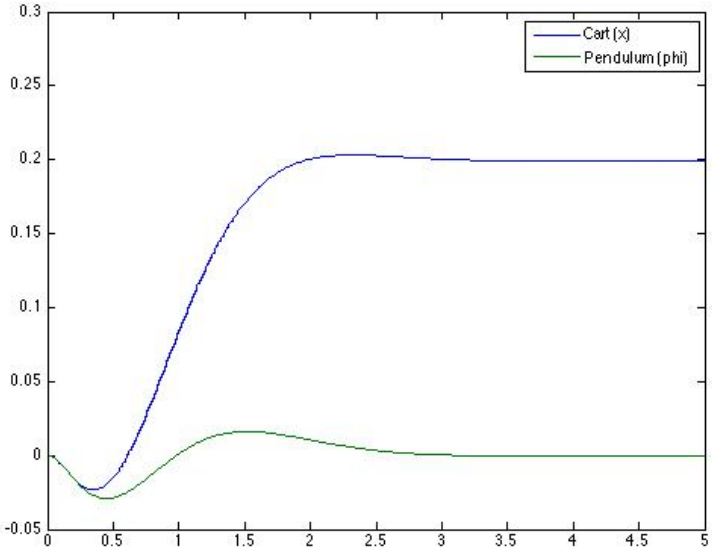


Figura 5.11: *Sistema amb el control lqr+ referencia*

Disseny d'un observador

El sistema de control dissenyat fins el moment s'ha basat assumint que tots els estats són mesurables. D'altra banda, aquesta hipòtesi no és vàlida en el nostre cas, per tant s'haurà d'incloure en el sistema algun mètode per tal d'estimar els valors dels estats no disponibles, a través de la informació mesurable de la planta.

Aquest mètode s'anomena observador, només es poden observar els estats només si es compleix la condició d'observabilitat analitzada anteriorment.

En aquest apartat es dissenya un observador d'ordre complet per tal d'obtenir la informació dels estats no mesurables ref.[8].

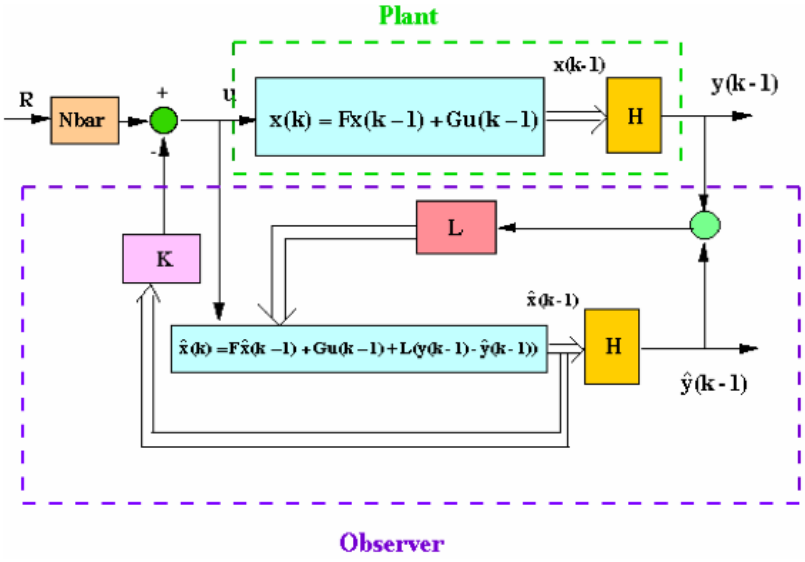


Figura 5.12: *Simulació de l'observador*

Passos pel disseny de l'observador:

- 1r pas. Per a obtenir la matriu L és necessari localitzar els pols del sistema sense l'observador [pols de (F-G*K)]. Un cop obtinguts els pols dels sistema, s'han de situar els pols de l'observador per tal que treballin més ràpid que el sistema sense l'observador.

$$poles = \begin{matrix} [0.9792 + 0.0198i] \\ [0.9792 - 0.0198i] \\ [0.9695 + 0.0014i] \\ [0.9695 - 0.0014i] \end{matrix}$$

$$P_{Lmatrix} = [-0.3 - 0.31 - 0.32 - 0.33]$$

- 2n pas. Obtenir la matriu L.

  $L = place(F', H', P_{Lmatrix})'$

$$L = \begin{bmatrix} 2.6290 & -0.0106 \\ 172.6195 & -1.3953 \\ -0.0121 & 2.6281 \\ -1.7531 & 172.7355 \end{bmatrix}$$

- 3r pas. Obtenir la resposta del sistema en conjunt, incloent l'observador.

$$Fce = \begin{bmatrix} F - G * K & G * K \\ zeros(size(F)) & (F - L * H) \end{bmatrix}; Gce = \begin{bmatrix} G * Nbar \\ zeros(size(G)) \end{bmatrix}$$

$$Hce = \begin{bmatrix} H & zeros(size(H)) \end{bmatrix}; Jce = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Tal i com s'ha estat fent en cadascun dels diferents apartats en aquest capítol, a través de la següent instrucció Matlab es simula el comportament de les matrius Gce,Gce,Hce i Jce, que en aquest cas són les matrius que descriuen el sistema incloent l'observador.

$[Y, X] = dlsim(Fce, Gce, Hce, Jce, U);$

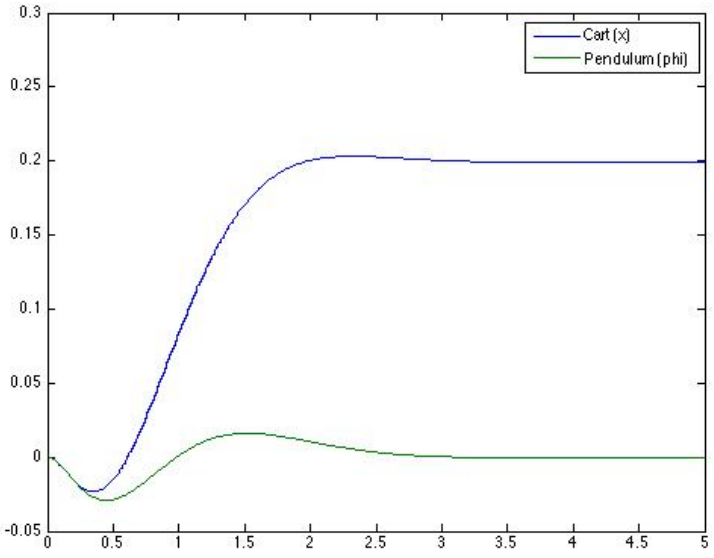


Figura 5.13: *Simulació observador*

Comparant les Figures (5.12) i (5.14) s'observa que la resposta del sistema, incorporant l'observador, no varia de la resposta sense l'observador; per tant l'observador dissenyat no altera el comportament del sistema.

Capítol 6

Obtenció del senyal de control i simulacions amb diferents condicions inicials

En aquest capítol es pretén aconseguir el senyal de control u específic per a cada instant de temps. Per tal d'aconseguir-ho, s'analitzarà el diagrama de blocs del sistema final i n'extraurem les equacions que defineixen u. El senyal de control u és el que s'aplica al sistema en un instant de temps (t), per tal de corregir l'error de la sortida en l'instant de temps (t-1). D'altra banda, també es simula el sistema sota diferents condicions inicials(c.i.). Aquestes simulacions es comparen amb el resultat del sistema amb c.i. nules, per tal que quedin més clar els diversos comportaments del pèndul quan es canvien les c.i. .

El codi complet per l'obtenció del senyal de control i les simulacions amb diferents c.i. es poden trobar a l'apèndix D.

6.1 Obtenció del senyal de control u

Analitzant el diagrama de blocs final del sistema (Figura 5.13), podem extreure el següent conjunt d'equacions per tal d'obtenir el senyal de control u.

$$\begin{aligned} u &= R \cdot N_{bar} - K \cdot x_s \\ x_s &= F \cdot x_s + G \cdot u + L \cdot (y - y_s) \\ x &= F \cdot x + G \cdot u \\ y &= x' \cdot H' \\ y_s &= x'_s \cdot H' \end{aligned}$$

Tal i com s'observa en l'anterior conjunt d'equacions, si es calcula l'error entre la sortida real i la sortida de l'observador obtenim els nous valors de les variables d'estat no mesurables. Un cop es coneixen totes les variables d'estat del sistema, s'obté el senyal de control u, que és el que s'envia al sistema per tal de corregir l'error anterior.

6.2 Simulacions amb diferents c.i.

A continuació es mostra una taula amb les diferents simulacions que es trobaran en aquesta secció. No obstant això, cal recordar que tot i que la simulació amb Matlab aconsegueixi estabilitzar sempre el pèndul, independentment de les c.i. que s'hi introdueixin, aquest model està linealitzat per actuar en una regió molt propera al punt estable neutral (± 0.2 rad/seg d'inclinació).

Simulació ID	inclinacio(t=0)	posicio(t=0)	inclinació(t=2.5)	posició(t=2.5)
1	0.179	0	0	0
2	0.179	0.2	0	0
3	0.179	-0.2	0	0
4	-0.179	0	0	0
5	-0.179	-0.2	0	0
6	-0.179	0.2	0	0
7	-0.179	0	-0.179	0.2
8	0.179	0	0.179	0.2
9	0	0	-1	0.2

El límit d'inclinació és de [-0.179,0.179], el qual ha estat escollit perquè per a inclinacions superiors el senyal de control u serà més gran que 255, que és la potència màxima que pot ser subministrada als motors del prototip.

Des d'aquest moment i fins el final del capítol es representen amb línies contínues el comportament del carro i la inclinació del pèndul, i amb línies discontínues el comportament del carro i la inclinació del pèndul amb c.i. nules i sense pertorbacions, per tal de poder apreciar amb més claretat la diferència de comportament en cada cas.

També trobarem en cada simulació els 10 primers senyals d'u a partir d'una pertorbació, ja que són els valors més rellevants que s'envien als motors per tal de rectificar l'error en t=(t-1).

6.2.1 Simulació 1:

En aquesta primera simulació el carro està inicialment inclinat 0.179 rads/seg.

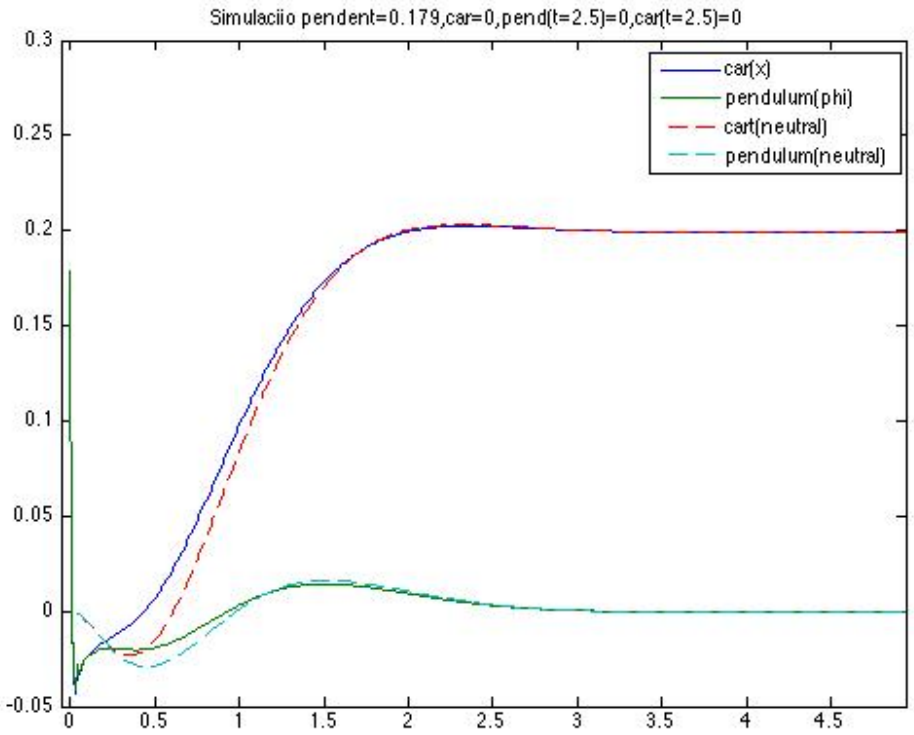


Figura 6.1: *Simulació 1*

Els deu primers valors d'u a partir de t=0.

$u_0 = -0.4240, u_1 = -171.6577, u_2 = 291.0885, u_3 = -167.8045, u_4 = 76.2389, u_5 = -32.2168, u_6 = 12.0721, u_7 = -5.0519,$

$u_8 = 1.4127, u_9 = -0.9132$

6.2.2 Simulació 2:

En la simulació 2, a part de la inclinació de la simulació anterior, es posa inicialment el carro en la posició 0.2, que és la posició final que ha d'aconseguir. Aquesta simulació té com a finalitat comprovar que tot i que el carro ja està en la posició desitjada, rectifica igualment la inclinació del pèndul.

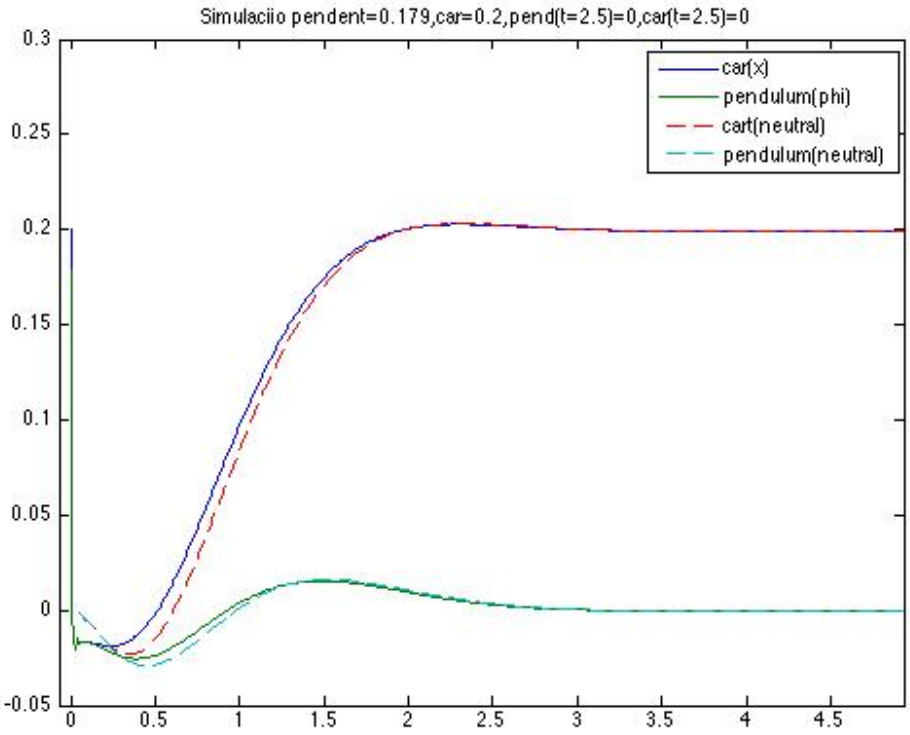


Figura 6.2: *Simulació 2*

Els deu primers valors d'u a partir de t=0.

$u_0 = -0.4240, u_1 = -81.1514, u_2 = 133.4528, u_3 = -74.5036, u_4 = 32.6137, u_5 = -13.3721, u_6 = 4.7733, u_7 = -1.9719,$

$u_8 = 0.4991, u_9 = -0.3385$

6.2.3 Simulació 3:

En aquesta simulació es pretén mostrar el recorregut que fa el carro per estabilitzar una inclinació inicial de 0.179 rads/seg, quan aquest es troba en una posició contrària a la desitjada.

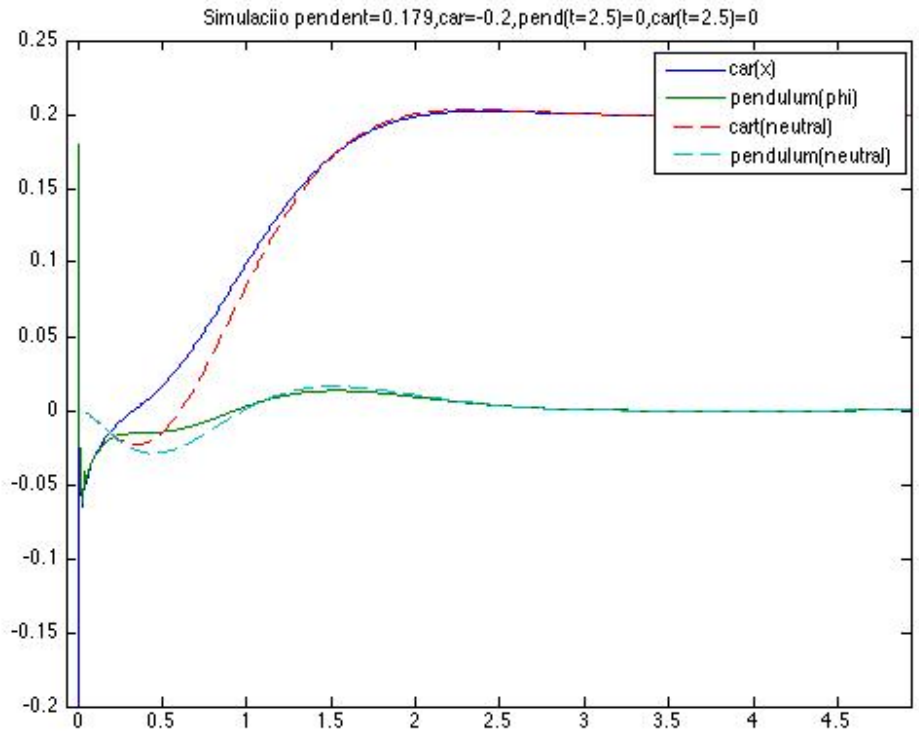


Figura 6.3: *Simulació 3*

Els deu primers valors d'u a partir de t=0.

$u_0 = -0.4240, u_1 = -262.1640, u_2 = 448.7243, u_3 = -261.1054, u_4 = 119.8641, u_5 = -51.0616, u_6 = 19.3709, u_7 = -8.1320,$

$u_8 = 2.3263, u_9 = -1.4879$

6.2.4 Simulació 4:

Aquesta simulació és molt pareguda a la simulació 1, però amb la inclinació canviada de signe. Aquesta simulació té la finalitat de mostrar els diferents moviments que realitza el pèndul amb una inclinació inicial negativa.

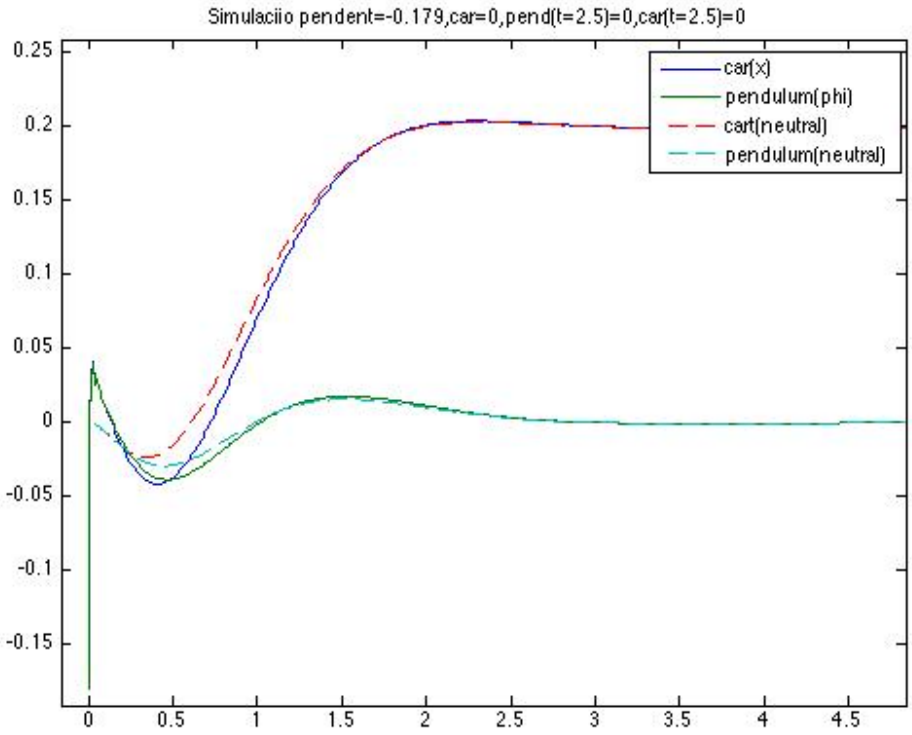


Figura 6.4: *Simulació 4*

Els deu primers valors d'u a partir de t=0.

$u_0 = -0.4240, u_1 = 170.8942, u_2 = -291.7717, u_3 = 167.1977, u_4 = -76.7734, u_5 = 31.7510, u_6 = -12.4730, u_7 = 4.7125,$

$u_8 = -1.6941, u_9 = 0.6866$

6.2.5 Simulació 5:

En la següent simulació, a part de tenir la mateixa inclinació que en la simulació anterior, també s'hi afegeix una posició del carro negativa. Podem apreciar, que quan la inclinació i la posició tenen el mateix signe, la resposta en les primeres dècimes de segon és més suau.

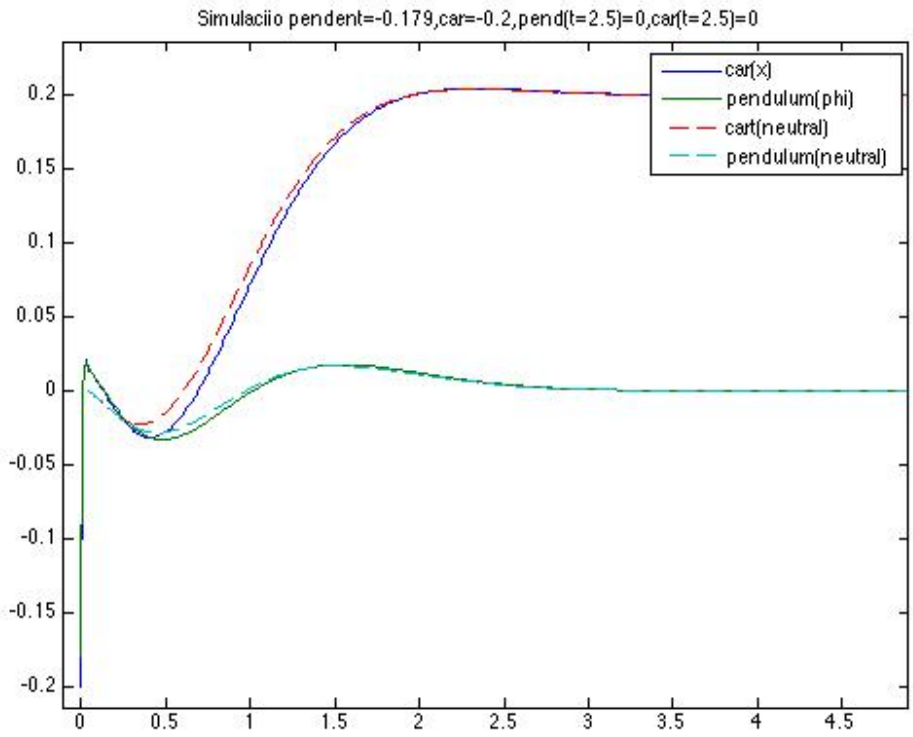


Figura 6.5: *Simulació 5*

Els deu primers valors d'u a partir de t=0.

$u_0 = -0.4240, u_1 = 80.3879, u_2 = -134.1359, u_3 = 73.8968, u_4 = -33.1482, u_5 = 12.9062, u_6 = -5.1742, u_7 = 1.6324,$

$u_8 = -0.7805, u_9 = 0.2324$

6.2.6 Simulació 6:

En aquesta simulació, a diferència de l'anterior, la posició del carro ja és la desitjada —0.2—, mentre que la inclinació és negativa. Com a resultat d'això, tal i com es mostrar en la següent figura, les primeres dècimes de segon tenen una intensitat de moviments més elevada que en la gràfica anterior.

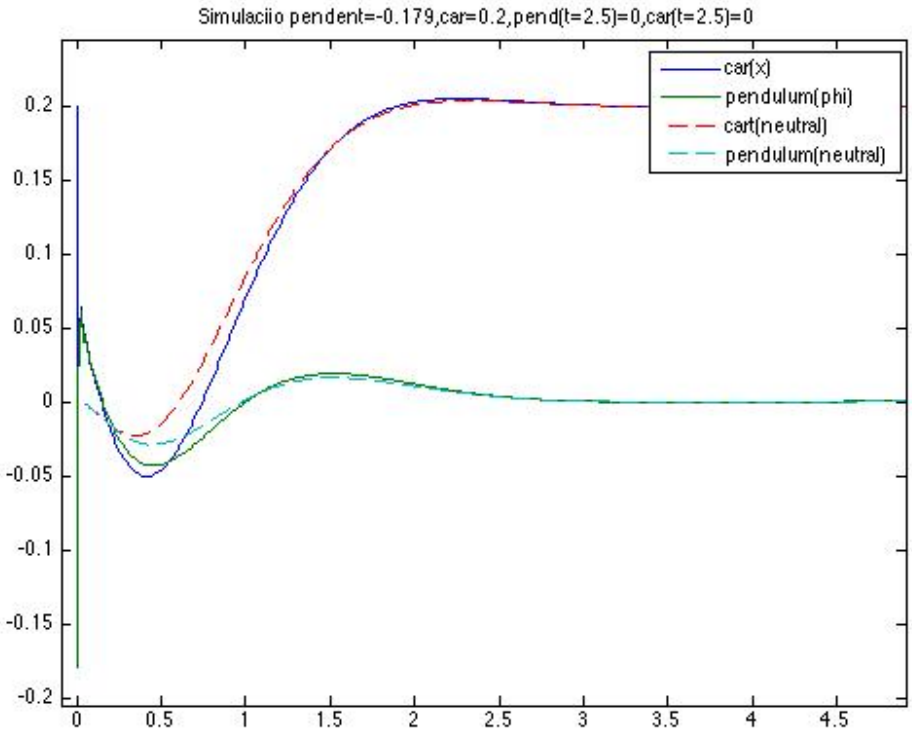


Figura 6.6: *Simulació 6*

Els deu primers valors d'u a partir de t=0.

$u_0 = -0.4240, u_1 = 261.4005, u_2 = -449.4074, u_3 = 260.4985, u_4 = -120.3985, u_5 = 50.5957, u_6 = -19.7718,$

$u_7 = 7.7925, u_8 = -2.6077, u_9 = 1.2613$

6.2.7 Simulació 7:

En aquesta simulació, deixant de banda les condicions inicials que ja s'han simulat anteriorment, també s'introdueixen pertorbacions en l'instant de temps t=2.5seg. En t=2.5 només es canvia la inclinació del pèndul amb la finalitat de veure amb més detall els moviments que realitza el carro, per tal de corregir la pertorbació i mantenir l'estabilitat.

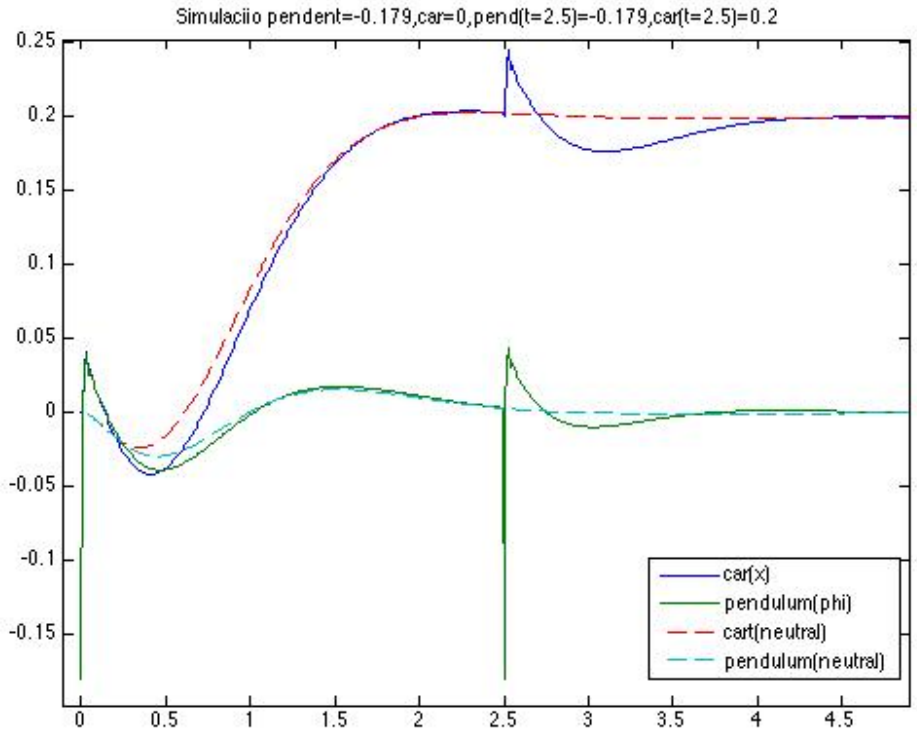


Figura 6.7: *Simulació 7*

Els deu primers valors d'u a partir de t=2.5.

$u_0 = -0.0042, u_1 = 173.2218, u_2 = -294.6897, u_3 = 169.3234, u_4 = -77.3261, u_5 = 32.3122, u_6 = -12.4012,$

$u_7 = 4.9268, u_8 = -1.5718, u_9 = 0.8044$

6.2.8 Simulació 8:

Aquesta simulació és igual que l'anterior, però amb la pertorbació en l'instant t=2.5seg. positiva.

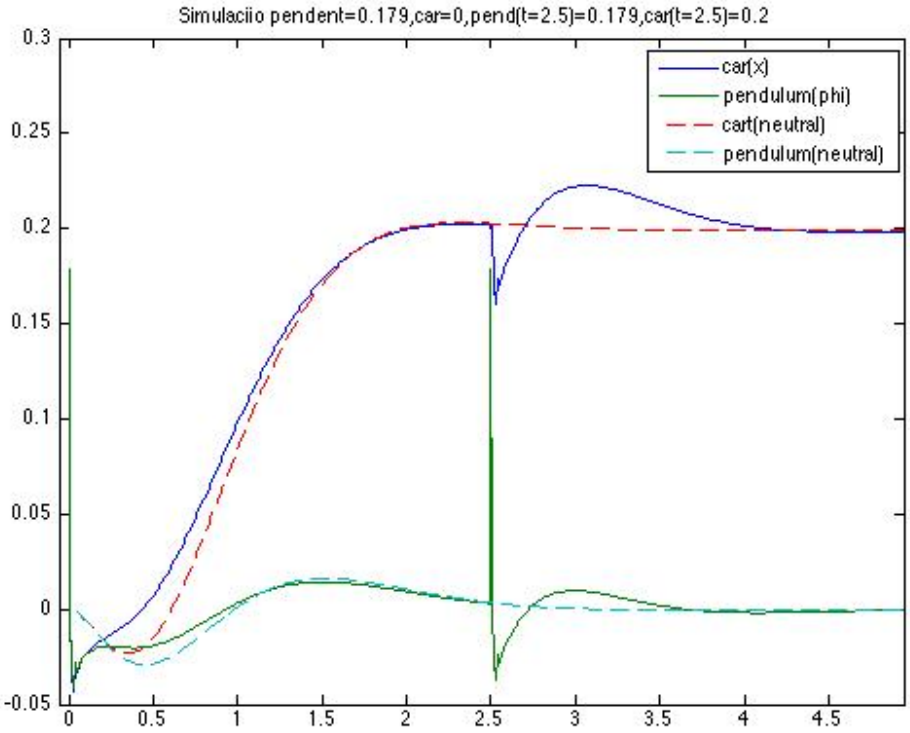


Figura 6.8: *Simulació 8*

Els deu primers valors d'u a partir de t=2.5.

$u_0 = -0.0038, u_1 = -169.3581, u_2 = 288.1965, u_3 = -165.6769, u_4 = 75.6793, u_5 = -31.6489, u_6 = 12.1409,$

$u_7 = -4.8353, u_8 = 1.5346, u_9 = -0.7948$

6.2.9 Simulació 9:

Aquesta simulació és un cas especial, per tal d'ilustrar que l'algoritme Matlab sempre estabilitzarà el sistema. Cal destacar que per aconseguir que el sistema sempre aconsegueixi assolir un estat estacionari, es sacrifica la restricció de sobrepic i el valor del senyal de control u és exageradament més gran del que es pot aplicar en el cas real.

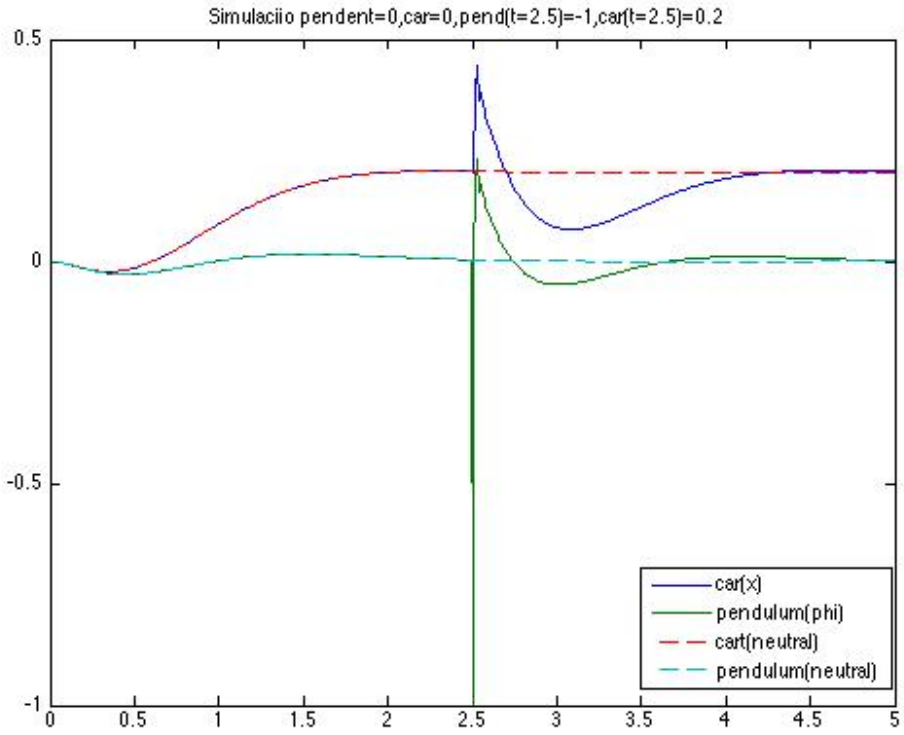


Figura 6.9: *Simulació 9*

Els deu primers valors d'u a partir de t=2.5.

$u_0 = -0.0040, u_1 = 958.7806, u_2 = -1.6313e+03, u_3 = 937.5835, u_4 = -428.2320, u_5 = 179.0127, u_6 = -68.6917,$

$u_7 = 27.3206, u_8 = -8.6967, u_9 = 4.4737$

Valoració de les simulacions

En totes les simulacions es compleixen les restriccions temporals i de sobrepic.

És especialment important observar el comportament del pèndul en les simulacions 7 i 8, són les més rellevants i són on millor es poden apreciar les diferents característiques del comportament. A més, si observem el valor u_1, en aquestes simulacions és on millor s'aprecia la direcció que pren el carro per rectificar una determinada inclinació.

Cal destacar que el valor d'u per t_0 o $t_{2.5}$ és molt petit, ja que pràcticament l'error en (t_0-1) i ($t_{2.5}$-1) és 0. Ara bé, si ens fixem en els valors de (t_0+1) i ($t_{2.5}$+1), es pot apreciar com el sistema ja ha fet *feedback* i aplicar un senyal de control proporcional a l'error obtingut en l'instant anterior.

Capítol 7

Algoritme de control cas real

En aquest capítol, s'implementa un algoritme de control en codi Arduino per tal de controlar el prototip utilitzat durant aquest treball.
Aquesta secció es divideix en tres subseccions: disseny de l'algoritme – on es comenten tots els passos i les diferències entre l'algoritme de les simulacions Matlab i el d'Arduino –, anàlisi de l'algoritme – on s'analitza el cost computacional en temps que suposarà l'execució de l'algoritme per al microcontrolador Arduino –, i finalment unes conclusions on es valoren els resultats obtinguts.

7.1 Disseny de l'algoritme de control

L'algoritme de control consisteix en la generació del senyal de control u per estabilitzar el sistema en temps real. L'estructura que mostra és molt similar a l'algoritme utilitzat en les simulacions del capítol anterior. Les principals diferències observades vers la simulació són detallades a continuació:

- 1r: S'ha de mesurar la variable d'inclinació. Per tant, s'ha d'afegir el codi referent a la lectura de l'acceleròmetre xip MinIMU-9.
- 2n: Es crea un *switch* simulant la taula creada en la secció 3.1 per la tranformació de les dades llegides per l'acceleròmetre a radiants, tal com s'explica a la secció 3.1.
- 3r: Configuració del senyal de control PWM.
- 4t: Traducció del codi Matlab a codi Arduino.

- 5e: Canviar la variable d'inclinació y[2][1] pels radiants obtinguts i enviar el senyal u al *OCR2B*.

Per tant, l'estructura final de l'algoritme de control implementat en Arduino consta de: una part de configuració on trobem els paràmetres que defineixen el xip min-IMU-9, la senyal PWM i el *timer* – utilitzat per generar les interrupcions internes de les funcions auxiliars –; una part on es declaren i es defineixen totes les matrius del sistema; una funció encarregada de llegir les dades del sensor periòdicament; una funció amb la finalitat de traduir les dades del sensor a radiants; i finalment el cos principal de l'algoritme on es troben les equacions que defineixen el senyal de control i posterioment envien aquest senyal als motors.
A l'apèndix E està descrita la totalitat del codi aplicat en el prototip de pèndul invertit.

7.2 Anàlisi de l'algoritme

En l'execució d'aquest algoritme és molt important el temps que tarda, ja que és en temps real. Així doncs, l'objectiu d'analitzar el temps d'execució de l'algoritme és comprovar que l'execució d'aquest no és més llarga que el propi temps de mostreig (ts=100ms).
Observant l'algoritme, s'aprecia que la part que el penalitza més són totes les operacions algebraiques que es realitzen per calcular les matrius del sistema, ja que totes es realitzen amb el tipus de dades *float*, i aquests, en el microprocessador Arduino, tenen un tamany de 4 bytes.
Per tal de saber el temps que tarden totes aquestes operacions algebraiques en executar-se es realitzen unes proves utilitzant el programa Atmel AVR Simulator. Aquestes proves consisteixen en executar una suma i multiplicació amb números reals i les seves corresponents assignacions. Tot seguit, es mira el codi *assembler* de l'execució anterior, on es compten 20 instruccions d'un cicle per a la suma i 20 més per a la multiplicació. L'algoritme té 61 multiplicacions tipus *float* i 47 sumes del mateix tipus. Per tant $(61 + 47) * 20 = 2160$cicles, i la freqüència del microcontrolador és de 16MHz. Si es divideixen els cicles totals entre la freqüència, s'obté el temps que triga l'algoritme en calcular les matrius del sistema. $2160/16MHz = 0.000135s = 135\mu s$.
Amb aquest temps s'observa que, el temps que triga l'algoritme en fer les operacions algeibraiques és molt inferior als 0.1seg de periode que té l'algoritme. Per tant queda demostrat que l'algoritme es suficientment ràpid com per a calcular la senyal de control amb les noves dades llegides cada cop.

7.3 Resultats

Després de la realització de diverses alternatives, s'aconsegueix una estabilització parcial del sistema utilitzan el controlador LQR. Es diu que l'estabilització és parcial perque no respont d'una manera fina, sinó que les reaccions per re-estabilitzar el sistema són un tant brusques.

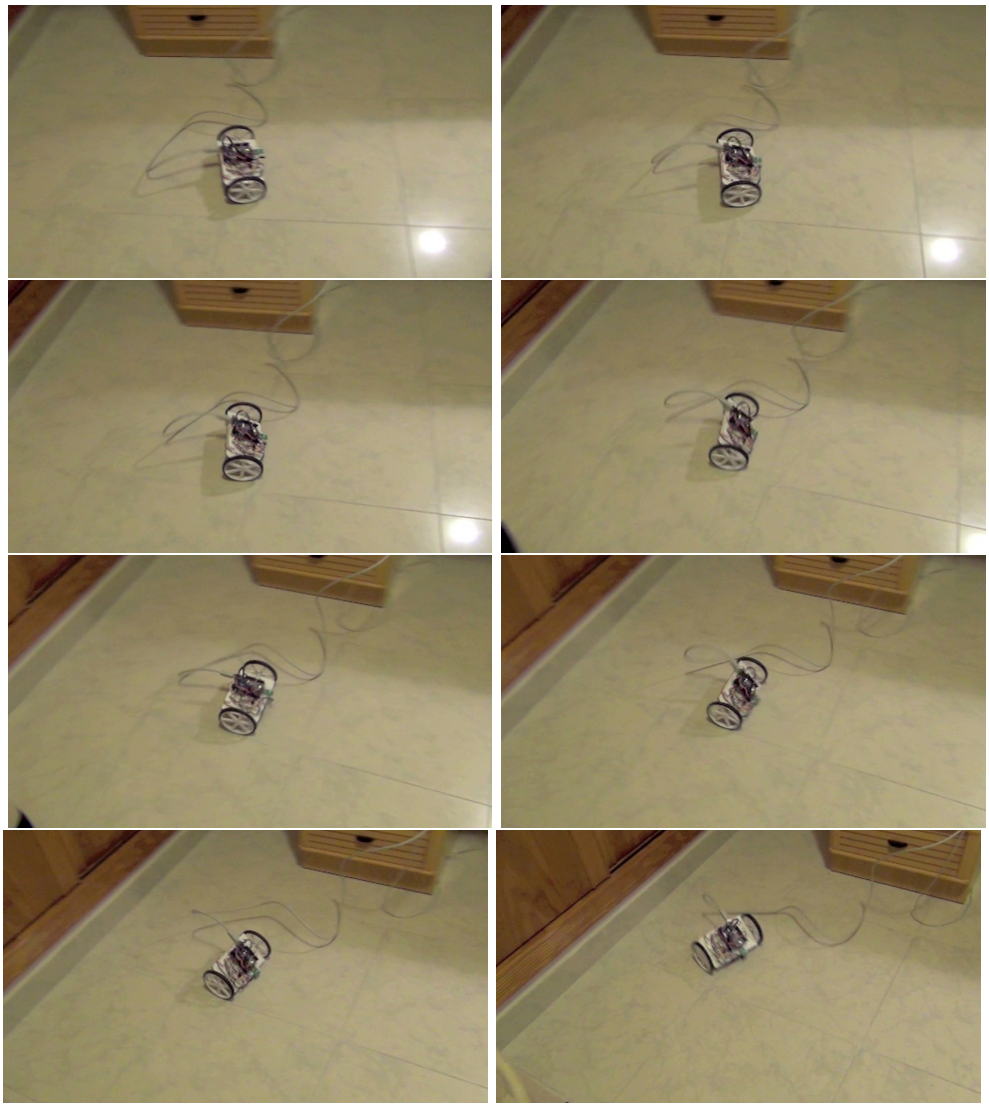


Figura 7.1: *Fotogrames de la demostració*

Capítol 8

Resultats i Conclusions

Durant el transcurs d'aquesta memòria, s'ha construït un prototip de pèndul invertit per tenir un cas real on poguéssim provar finalment els algoritmes de control. Posteriorment, s'ha definit un model matemàtic basant-se amb les forces que actuen sobre el sistema, per tal d'obtenir unes equacions que descriguin el comportament del sistema, que són la funció de transferència i la representació amb variables d'estat. A continuació, s'han dissenyat dos tipus de control diferents, un PID per a la representació del sistema, amb funció de transferència, i un LQR per a la representació amb variables d'estat, amb l'objectiu d'estabilitzar el sistema. Tot seguit, s'han realitzat diverses simulacions amb el disseny final, implementat per un controlador LQR, per tal d'obtenir els diferents senyals de control que serien enviats als motors davant diferents condicions inicials i pertorbacions. Per últim, s'ha programat el control final, basat en una representació amb variables d'estat i un disseny de control LQR, al prototip de pèndul invertit.

El plantejament inicial del projecte fou l'estudi de control d'un pèndul invertit, ara bé, ens vam adonar que per dur a terme aquest control no disposàvem de cap prototip real . Així doncs, un dels principals objectius previs al control fou la construcció d'un prototip de pèndul invertit. El disseny donat va ser la construcció d'un *segway*, on la mateixa estructura del carro forma part del pèndul invertit.

El primer problema a superar fou el cost del prototip. En no disposar d'un pressupost elevat per comprar el prototip ja muntat, es decideix construir un propotip econòmic.

D'altra banda, quan ohm decideix construir un sistema real com el pèndul invertit, es trobar en alguns problemes de construcció, com per exemple el mal funcionament dels sensors o la bateria.

Finalment, un cop detectat l'origen dels problemes, s' opta per comprar unes peces noves per tal de garantir el funcionament.

Un cop construït el prototip i comprovat el seu correcte funcionament, es passa a treballar amb el 2n principal objectiu del projecte, el control del pèndul invertit.
Inicialment es proposa un disseny PID per al controlador. Un cop realitzades les simulacions d'aquest, ens vam donar compte que el mòdul que controla l'estabilitat del prototip presenta un sobrepic massa elevat i que quan s'afegia el mòdul per controlar la posició del carro perdia l'estabilitat passats els 10 seg. de funcionament. Després d'evaluar aquesta resposta del sistema, es decideix canviar el disseny del controlador per un disseny LQR. Aquest disseny inicialment ens donava l'avantatge que ens permetia treballar amb variables d'estat, la qual cosa significava que podíem tenir controlades les variables d'inclinació i posició del carro. Les simulacions realitzades amb aquest controlador compleixen amb els requisits d'estabilitat del sistema. Seguidament, es desenvolupa un algoritme per tal d'aconseguir un senyal de control, capaç d'estabilitzar el sistema davant diferents condicions inicials. Finalment, s'implementa aquest últim algoritme en el prototip real, però els resultats no foren els esperats. Després de diferents intents i modificacions, l'algoritme de control només fou capaç d'estabilitzar parcialment el sistema.
Un dels motius que pot explicar el mal funcionament de l'algoritme de control seria el model matemàtic aplicat o la baixa robustesa del disseny de control. El model matemàtic aplicat es basa en una visió general del comportament físic del pèndul invertit, el qual reflecteix el model general teòric de comportament, que pot presentar certes diferències amb el cas pràctic real que descriu el comportament físic del pèndul. Aprofundint més en les bases teòriques dels controladors LQR, es va observar que aquests tipus de controlador presenten una baixa robustesa, fet que significa que es garantitza el funcionament d'aquests controladors només quan hi ha poca diferència entre el model matemàtic utilitzat i el model físic real.
Com a línia futura i basada en l'experiència adquirida durant la construcció d'aquest pèndul invertit senzill, es proposa: per una banda construir un nou prototip de pèndul invertit utilitzant uns motors de més qualitat i una estructura del carro més robusta; per altra banda aplicar un control híbird composat de un controlador PID per al problema de *swing up* i un controlador LQR per al problema d'estabilitat local.

Bibliografia

[1] Web oficial arduino: *http://www.arduino.cc/es/*

[2] Web oficial ATMEL: *http://www.atmel.com/devices/atmega328p.aspx?tab=documents*

[3] Datasheet l293d: *http://idmax.free.fr/Aide/Stepper/l293.pdf*

[4] l293d exemples d'aplicació: *http://galia.fc.uaslp.mx/~cantocar/microcontroladores/SLIDES_8051_PDF/11_L293.PDF*

[5] Pèndul de K. Ogata "Dinámica de Sistemas". Ed. Prentice Hall Hispanoamerica- na, pp. ISBN 0 -13-880385-4.

[6] Model de pèndul invertit *http://www.engin.umich.edu/group/ctm/examples/pend/invpen.html*

[7] Ogata K. Ingeniera de Control Moderna. Ed. Prentice Hall International, New Jersey. 1974.

[8] Ingeniería de control: Modelado y control de sistemas dinàmicos. Ed Ariel Ciencia. 2003. ISBN: 84-344-8055-7

[9] Feedback System. An introduction for Scientists and Engineers. Astrom and Murray. ISBN-13: 978-0-691-13576-2 MinIMU-9 Informació tècnica sobre el xip MinIMU-9 *http://www.pololu.com/catalog/product/1264*

APÈNDIX

APÈNDIX A

Model del motor procés Arduino

Aquest procés Arduino té com a objectiu recol·lectar les dades enviades pel sensor IR, que són el número de radis cada 0.1seg, per tal que el procés Matlab que es troba en l'apèndix següent, pugui extreure un model matemàtic dels motors. Per obtenir aquestes dades, aquest procés força el motor amb uns rangs d'entrades que van des de la mínima fins la màxima potència. Aquests rangs d'entrades s'intercalen amb entrades de potència 0, per poder observar amb més detall la resposta del motor.

```
#include <avr/io.h>
#include <avr/interrupt.h>
#define PWM_ 3
#define PWM2_ 3
#define ENABLE 12
#define pot 0
#define invertir_pin 7
int intAnValue=0;
int ncops=0;
const byte AnInputPin=0; // leer del puerto analogico 0.
void setup()
{
pinMode(PWM_, OUTPUT);
pinMode(PWM2_, OUTPUT);
```

```
pinMode(invertir_pin, OUTPUT);

Serial.begin(9600);

// initialize Timer1

digitalWrite(ENABLE, HIGH);

digitalWrite(invertir_pin, LOW);

//timer 2 es de 8 bits

//wgm = 011 is pwm

// COM2A AND COM2B BIT TO 10 = non-inverted PWM

//CS BITS TO 111 = prescaler to divide the clock by 1024.

TCCR2A = _BV(COM2A1)|_BV(COM2B1)|_BV(WGM21)|_BV(WGM20);

TCCR2B = _BV(CS22)|_BV(CS21)|_BV(CS20);

OCR2A= 0; // output 11

OCR2B = pot;//output 3 ;)

attachInterrupt(0, sensorIR, RISING);

////////////

//////// TIMER 1 SET UP//////////

//timer 1 16 bits

// initialize Timer1

TCCR1A = 0; // set entire TCCR1A register to 0

TCCR1B = 0; // same for TCCR1B

// set compare match register to desired timer count:

//OCR1A = 15624; -> per interrupcio cada 1 segon

// (#timer counts = OCR1A)

// target time = temps cada quant es produira una interrupcio

//timer resolution = 1/16Mhz/1024 ; 1024 del preescaler

// (timercounts + 1) = targettime/timerresolution

// timercounts = (0.1/(6.4 * 10^-5)) - 1

OCR1A = 1561;

//els registres TCCRxA i TCCRxB son els encarregats de fer la

// configuracio del timer.

// turn on CTC mode:
```

```
TCCR1B |= (1 « WGM12);

// Set CS10 and CS12 bits for 1024 prescaler:

TCCR1B |= (1 « CS10);

TCCR1B |= (1 « CS12);

// enable timer compare interrupt: amb preescales de clk/1024

TIMSK1 |= (1 « OCIE1A);

// enable global interrupts:

////////////////////////////

sei();

}

void loop(){

ncops=0;

OCR2B= 0;

delay(3000);

OCR2B= 50;

delay(3000);

OCR2B= 0;

delay(3000);

OCR2B= 100;

delay(3000);

OCR2B= 0;

delay(3000);

OCR2B= 150;

delay(3000);

OCR2B= 0;

delay(3000);

OCR2B= 200;

delay(3000);

OCR2B= 0;

delay(3000);

OCR2B= 250;
```

```
delay(3000);

}

void sensorIR(){//interrupcio externa del sensor IR

ncops=ncops+1;

}

ISR(TIMER1_COMPA_vect)//interrupcio interna cada 0.1segs


Serial.println((int)ncops);

ncops=0;

}
```

APÈNDIX B

Model del motor procés Matlab

Aquest procés Matlab té com a objectiu retornar una funció de transferència que descrigui el funcionament dels motors, basant-se en les dades obtingudes en el procés Arduino de l'apèndix A.

```
    nomFitxer=input('Open file: ', 's');
[path, nom, ext, vers] = fileparts(nomFitxer);
A=load(nomFitxer);
tsample = 0.1025 //Sampling time
bottom = 50000 //Substract the offset
y = A(:,end)./2; //Propeller passes ales times each turn
u = (A(:,1)-50000)./1000; // Divide by 1000 in order to get lower input
z = iddata(y,u,tsample);
z.InputName = 'Potencia';
z.OutputName = strcat('Velocitat angular (',nom,')');
z.TimeUnit = 'Seg';
z.InputUnit = 'Passos';
z.OutputUnit = 'Rads/s';
figure(1);
idplot(z);
//ze = z(100:1400);
//figure(2);
//idplot(ze);
```

```
//zd = detrend(ze);

//figure(3);

//idplot(zd);

zd = z;

th = arx(zd,[1 1 1])

figure(4);

compare(zd, th);

th = sett(th, tsample);

[numd, dend]=th2tf(th);

printsys(numd, dend, 'z');

[numc, denc]=d2cm(numd, dend, tsample,'matched');

printsys(numc, denc, 's');
```

APÈNDIX C

PID per a TF

Aquest codi treu el *feedback* de la planta sense controlador i amb controlador. També mostra el lloc de les arrels del sistema amb controlador i sense, per poder veure la diferència d'estabilitat dels dos sistemes. Per finalitzar, també simula la posició del carro.

```
clc
clear
syms s
%%%%%%%%%%%%%%%%%%
%%%%%% PARAMETROS DEL SISTEMA
%%%%%%%%%%%%%%%%%%
M = 0.02;
m = 0.250;
l = 0.035;
b = 0.1;
i = m * ((l^2/12) + l)%0.006; %0.006;
g = 9.8;
%%%%%%%%%%%%%%%%%%
%%%%%%% MODEL AMB FUNCIO DE TRANSFERENCIA
%%%%%%%%%%%%%%%%%%
q = (M + m) * (i + m * l^2) - (m * l)^2;
num = [m * l/q0]
den = [1b * (i + m * l^2)/q - (M + m) * m * g * l/q - b * m * g * l/q]
num_cart = [((i + m * l^2)/q), 0, -(m * g * l/q)];
den_cart = [1, (b * (i + m * l^2))/q, -((M + m) * m * g * l)/q, -b * m * g * l/q];
% la instrucci? impulse ens mostra la sortida del
% sistema davant una entrada escal?
%%%%%%%%%%%%%%%%%%
%%%%%%%% MODEL MOTOR %%%%%%%%%%%%%%%%%%
num2 = [1.4063];
den2 = [174.2386];
%%%%%%%%%%%%%%%%%%
%%%%%%%% MODEL PLANTA
%%%%%%%%%%%%%%%%%%
%esta nova figura es amb el sistema compler,
%i mean mes el model del motor
num_s = conv(num2, num);
den_s = conv(den2, den);
pend = tf(num_s, den_s)
```

```
    %% figure(1)
fs = feedback(pend, 1)
step(fs)
title('feedback_closed_loop')
%%figure(2)
[num,den]=tfdata(pend,'v');
t=0:0.01:25;
impulse(num, den, t)
axis([025 − 2060])
title('open_loop')
%%figure(3)
rlocus(pend)
title('RootLocus')
%%figure(4)
z = [-3 -4];%%zeros
p = 0;%%polos
k = 1;%%ganancia
C = zpk(z,p,k)

rlocus(C ∗ pend)
title('RootLocuswithPIDController')
%%
    [k, poles] = rlocfind(C ∗ pend)%retorna la ganancia corresponent a un punt en
%en el lloc de les arrels(root locus)
K = 100;
T = feedback(pend, K ∗ C);
impulse(T)
title('Response of Pendulum Angle to an Impulse Disturbance under PID Control');

    %% figure(6)
num_cart = [((i + m ∗ l^2)/q), 0, −(m ∗ g ∗ l/q)];
den_cart = [1, (b ∗ (i + m ∗ l^2))/q, −((M + m) ∗ m ∗ g ∗ l)/q, −b ∗ m ∗ g ∗ l/q];


    P_cart = tf(num_cart, den_cart)

T2 = feedback(1, pend ∗ C) ∗ P_cart;

t = 0 : 0.01 : 5;

impulse(T2, t);

title('Response of Cart Position to an Impulse Disturbance under PID Control');
```

APÈNDIX D

Codi simulacions

El següent codi Matlab calcula el senyal de control u per a cada instant de temps partint d'unes condicions inicials. Per cada instant de temps també es guarda el resultat de les sortides, per poder obtenir una gràfica amb el comportament del sistema.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% IMPLEMENTACIO DEL ALGORITME
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Se inicializan las variables
yys=[0, 0] ;
xxs=[0, 0, 0, 0];
xx=[0, 0, 0, 0];
yy=[0,0.179];
uu =0;
i=1;
R=zeros(1,500);
for i=1:500
R(1,i)=0.2;
%R=zeros(1,100);
end


for i=1:length(R)
```

```
%- tomar datos ---

x = xx(i,:)';

y = yy(i,:)';

ys=yys(i,:)';

xs=xxs(i,:)';

%- calculos ----

u = R(i) * Nbar - (xs' * K') %- control ----

xs = F * xs + G * u + L * (y - ys); %- control ? observer ---

x = (F * x + G * u); %- planta ----

y = x' * H'; %- planta ----

ys = xs' * H'; %- control observer ---

%- anotar resultados

xx(i + 1, :) = x(:)';

yy(i + 1, :) = y(:)';

uu(i + 1) = u;

xxs(i + 1, :) = xs(:)';

yys(i + 1, :) = ys(:)';

end

t=0:0.01:5;

hold on;

figure(5)

plot(t,yy,t,Y,'-');

legend('car(x)', 'pendulum(phi)','cart(neutral)','pendulum(neutral)');

title('Simulaciio pendent=-0.179')

hold off;
```

APÈNDIX E

Controlador cas real

A continuació es pot veure el codi que implementa el controlador en el microprocessador Arduino.

```
include <Wire.h>
#include <LSM303.h>
#include <avr/io.h>
#include <avr/interrupt.h>

#define PWM3
#definePWM23
#defineENABLE12
#definepot0
#defineinvertirpin7
//accelerometre
LSM303compass;
//Vin->5v
//GND->GND
//SCL->analogpin5
//SDA->analogpin4escriuillegeix
intvalorx=0;
floatrads;
//----------MATRIUSDELSISTEMA----------
floatK[1][4]={{-2.1273,-2.5345,16.2812,5.2711}};
floatF[4][4]={{1,0.0100,0,0},{0,0.9962,0.0032,0},{0,0,10005,0},{0,-0.0037,0.0975,1.0005}};
floatu=0;
floatxs[4][1]={{0},{0},{0},{0}};
floatx[4][1]={{0},{0},{0},{0}};
floatG[4][1]={{0.002},{0.0382},{0},{0.0368}};
floatL[4][2]={{2.6290,-0.0106},{172.6195,-1.3953},{-0.121,2.6281},{-1.7531,172.7355}};
floatH[2][4]={{1,0,0,0},{0,0,1,0}};
floaty[2][1]={{0},{rads}};
floatys[2][1]={{0},{0}};
floatNbar=-2.12;
//----------
//---------matriusauxiliarsperclacu
floatFxs[4][1]={{0},{0},{0},{0}};
floatFx[4][1]={{0},{0},{0},{0}};
floatGu[4][1]={{0},{0},{0},{0}};
floatLyys[4][1]={{0},{0},{0},{0}};
```

```
floatyys[2][1]={{-2},{0}};
floatKxs=0;
//-------------------------------------------------------------------

voidsetup(){
//////////////////////
////////ACCSETUP
pinMode(PWM, OUTPUT);
pinMode(PWM2, OUTPUT);
pinMode(invertir_pin, OUTPUT);
Serial.begin(9600);
digitalWrite(ENABLE, HIGH);
digitalWrite(invertirpin, LOW);
Wire.begin();
compass.init();
compass.enableDefault();
////////////////////////////////
//////// TIMER 1 SET UP//////////
//timer 1 16 bits
// initialize Timer1
cli(); // disable global interrupts
TCCR1A= 0; // set entire TCCR1A register to 0
TCCR1B = 0; // same for TCCR1B

// set compare match register to desired timer count:
//OCR1A = 15624; -> per interrupcio cada 0.1 segon
// (timer counts = OCR1A)
// target time = temps cada quant es produira una interrupcio
//timer resolution = 1/16Mhz/1024 ; 1024 del preescaler
// (timercounts + 1) = targettime/timerresolution
// timer counts = (0.1/(6.4 * 10^-5)) - 1
OCR1A = 1562;
//els registres TCCRxA i TCCRxB son els encarregats de fer la
// configuracio del timer.
// turn on CTC mode:
TCCR1B |= (1 << WGM12);
// Set CS10 and CS12 bits for 1024 prescaler:
TCCR1B |= (1 << CS10);
TCCR1B |= (1 << CS12);
// enable timer compare interrupt: amb preescales de clk/1024
TIMSK1 |= (1 << OCIE1A);
// enable global interrupts:
////////////////////////////

sei();

/////_____CONFING_PWM----------
TCCR2A =_BV(COM2A1) |_BV(COM2B1) |_BV(WGM21) | _BV(WGM20);
TCCR2B =_BV(CS22)|_BV(CS21)|_BV(CS20);
OCR2A = 0;//output11
OCR2B = 0;//output3;)

}

intllegir(){
intx;
x = compass.a.x;
returnx;
}
```

```
// ==========================
// === FUNACC2RAD ========
// ==========================
voidacc2rad(intvalor_x2){
intaccx = valor_x;

if((546 > accx)(accx > 462)){
rads = 0.546;
}
elseif(accx >= 546){
rads = 0.546;
}
elseif((462 > accx)(accx > 322)){
rads = 0.467;
}
elseif((322 > accx)(accx > 303)){
rads = 0.393;
}
elseif((303 > accx)(accx > 248)){
rads = 0.323;
}
elseif((248 > accx)(accx > 182)){
rads = 0.255;
}
elseif(182 > accx > 135){
rads = 0.189;
}
elseif((135 > accx)(accx > 84)){
rads = 0.125;
}
elseif((84 > accx)(accx > 10)){
rads = 0.062;
}
elseif((10 > accx)(accx > -10)){
rads = 0.0;
}
elseif((-10 > accx)(accx > -82)){
rads = -0.0625;
}
elseif((-82 > accx)(accx > -130)){
rads = -0.125;
}
elseif((-130 > accx)(accx > -198)){
rads = -0.189;
}
elseif((-198 > accx)(accx > -264)){
rads = -0.255;
}
elseif((-264 > accx)(accx > -315)){
rads = -0.323;
}
elseif((-315 > accx)(accx > -382)){
rads = -0.393;
}elseif(accx <= -315){
rads = -0.393;
}
}//FIACC2RAD

voidloop(){
compass.read(); //5muli3sum
```

```
Kxs=K[1][1]*xs[1][1]+K[1][2]*xs[2][1]+K[1][3]*xs[3][1]+K[1][4]*xs[4][1];
u=0.2*Nbar-Kxs;

if(u > 255){
u = 255;
Serial.print(- - -- >><< if1 : ");
}
elseif(u<-255){
Serial.print(- - -- >><< if2 : ");
u=-255;
}
if(u < -1){
//digitalWrite(invertir_pin, HIGH);
u=u+255;
}
elseif(u < 1){
digitalWrite(invertir_pin, LOW);
}
OCR2B = u;
Serial.print(- - -- >><< u : ");
Serial.println((float)u);
//2rest
yys[1][1]=y[1][1]-ys[1][1];
yys[2][1]=y[2][1]-ys[2][1];
for(int i=1;i<5;i++){//44muli21sum
Fxs[i][1]=F[i][1]*xs[1][1]+F[i][2]*xs[2][1]+F[i][3]*xs[3][1]+F[i][4]*xs[4][1];
Fx[i][1]=F[i][1]*x[1][1]+F[i][2]*x[2][1]+F[i][3]*x[3][1]+F[i][4]*x[4][1];
Gu[i][1]=G[i][1]*u;
Lyys[i][1]=L[i][1]*yys[1][1]+L[i][2]*yys[2][1];
}
for(int i=1;i<5;i++){//12sum
xs[i][1]=Fxs[i][1]+Gu[i][1]+Lyys[i][1];
x[i][1]=Fx[i][1]+Gu[i][1];
}

//12muli9sum
y[1][1]=H[1][1]*x[1][1]+H[1][2]*x[2][1]+H[1][3]*x[3][1]+H[1][4]*x[4][1];
//y[2][1]=H[2][1]*x[1][1]+H[2][2]*x[2][1]+H[2][3]*x[3][1]+H[2][4]*x[4][1];
y[2][1]=rads;
ys[1][1]=H[1][1]*xs[1][1]+H[1][2]*xs[2][1]+H[1][3]*xs[3][1]+H[1][4]*xs[4][1];
ys[2][1]=H[2][1]*xs[1][1]+H[2][2]*xs[2][1]+H[2][3]*xs[3][1]+H[2][4]*xs[4][1];
/ * Serial.print("A");
Serial.print("X : ");
Serial.println((int)valor_x);
* /
Serial.print(- - - - - - - - - - - - > RADS_2");
Serial.println((float)rads);
delay(100);

}

ISR(TIMER1_COMPA_vect)
{
valor_x = llegir();
acc2rad(valor_x);
}
```

APÈNDIX F

LQR per a un disseny de pèndul invertit amb variables d'estat

El següent procés Matlab té com a objectiu discretitzar el sistema, fer una anàlisi d'observabilitat, disseny d'un control òptim LQR aplicant-li una entrada de referència, i finalment dissenyar un observador d'ordre reduït.

```
clc
M = 0.02;
m = 0.250;
b = 0.1;
l = 0.035;
i = m*((l^2/12) + l)%0.006;
g = 9.8;
p = i*(M+m) + M*m*l^2; %denominadorperlesmatriusAyB
A = [0100;
0 - (i + m*l^2)*b/p(m^2*g*l^2)/p0;
0001;
0 - (m*l*b)/pm*g*l*(M+m)/p0]
B = [0;
(i + m*l^2)/p;
0;
m*l/p]
C = [1000;
0010]
D = [0;0]

%%%%%%%%%%%%%%%%%%%%%%%%
% DISCRETITZACIî DEL SISTEMA
%%%%%%%%%%%%%%%%%%%%%%%%
Ts=1/100; % freqncia de muestreig
[F,G,H,J] = c2dm(A,B,C,D,Ts,'zoh')
%%%%%%%%%%%%%%%%%
% ANALISSIS D'OBSERVABILITAT
%%%%%%%%%%%%%%%%%
% 'Controlabilidad.'
co = ctrb(F,G);
Controllability = rank(co)
% 'Observabilidad. '
ob = obsv(F,H);
```

$Observability = rank(ob)$

%%%%%%%%%%%%%%%
% DISSENY DEL LQR
%%%%%%%%%%%%%%%
%disp 'Ahora buscaremos el controlador K'
figure(1)
$T = 0 : 0.01 : 5; U = 0.2 * ones(size(T));$
$x = 5$; % factor de ponderaciî pER la posici del carro
$y = 0.1$; % factor de ponderaci paer l'angle del pndul
$Q = [x000;$
0000;
$00y0;$
0000];
$R = 1;$
$K = dlqr(F, G, Q, R)$
$[Y, X] = dlsim(F - G * K, G, H, J, U);$
stairs(T,Y)
legend('Cart (x)','Pendulum (phi)')

%%%%%%%%%%%%%%%
% ENTRADA DE REFERENCIA
%%%%%%%%%%%%%%%
$Nbar = -2.12;$
figure(2)
$[Y, X] = dlsim(F - G * K, G * Nbar, H, J, U);$
stairs(T,Y)
legend('Cart (x)','Pendulum (phi)')

APÈNDIX G

Diagrames de Gantt

En aquest apèndix trobem informació sobre la planificació inicial, les reunions amb el director del projecte i la planificació final.

Planificació inicial

tasques	Dara inici	Duració	
Planificació del projecte	8-Feb-12	5	13-Feb-12
Estudi de l'estat de l'art	20-Feb-12	40	31-Mar-12
Programació prototip ja montat	30-Mar-12	40	9-May-12
Model del sistema	2-May-12	20	22-May-12
Simulacions	22-May-12	40	1-Jul-12
Escriure la memoria	2-Jul-12	30	1-Aug-12
Proba real	1-Sep-12	40	11-Oct-12
Correcions memoria	29-Oct-12	15	13-Nov-12
Presentació	20-Nov-12	10	30-Nov-12
Entregar PFC	6-Jan-13	5	11-Jan-13

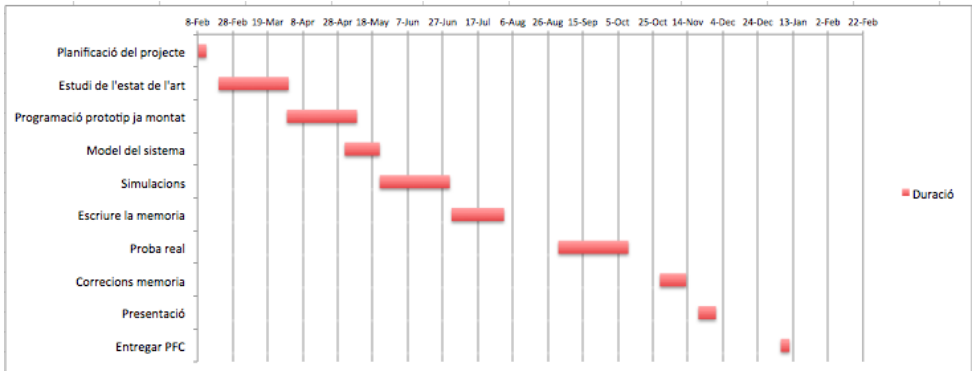


Figura G.1: *Diagrama de Gantt planificació inicial*

Reunions Director			
Tasques	Data inici	Duració	Data final
Reunió 1	20-Feb-12	1.3	20-Feb-12
Reunió 2	12-Mar-12	1.4	12-Mar-12
Reunió 3	03-Jun-12	1.45	03-Jun-12
Reunió 4	10-Jun-12	2	10-Jun-12
Reunió 5	17-Jun-12	2	17-Jun-12
Reunió 6	23-Jun-12	1.45	23-Jun-12
Reunió 7	24-Jul-12	1.4	24-Jul-12
Reunió 8	18-Sep-12	2	18-Sep-12
Reunió 9	25-Sep-12	1.4	25-Sep-12
Reunió 10	27-Sep-12	2.3	27-Sep-12
Reunió 11	04-Oct-12	2	04-Oct-12
Reunió 12	09-Oct-12	1.5	09-Oct-12
Reunió 13	23-Oct-12	2.3	23-Oct-12
Reunió 14	06-Nov-12	1.3	06-Nov-12
Reunió 15	22-Nov-12	1.3	22-Nov-12
Reunió 16	04-Dec-12	3	04-Dec-12
Reunio 17	17-Dec-12	2	17-Dec-12
reunio 18	8-Jan-13	2	08-Jan-13
reunio 19	22-Jan-13	2	22-Jan-13

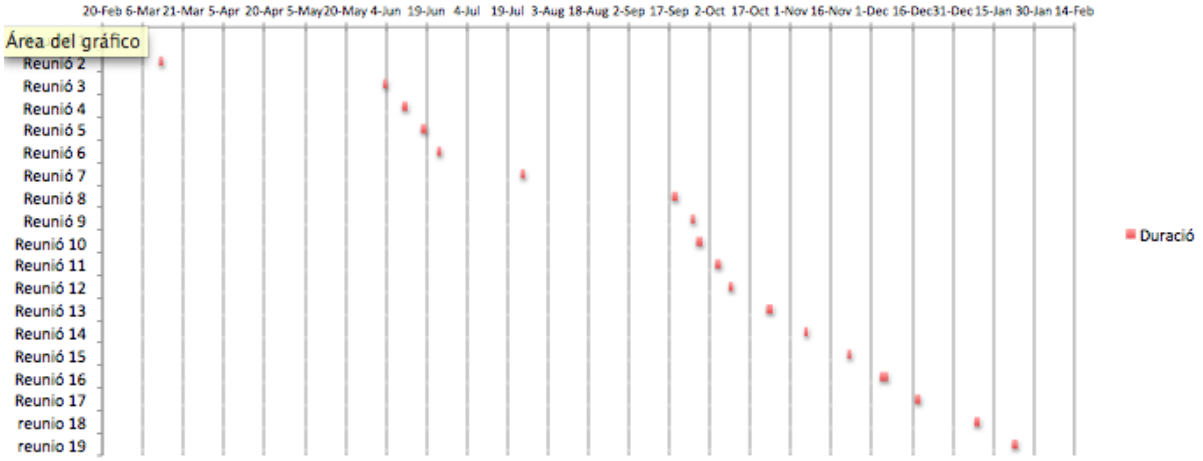


Figura G.2: *Diagrama de Gantt reunions director*

Tema		Data Inici	Duració	Data Final
TEMA1			5	
	Introducció.	26-Nov-12	5	27-Nov-12
TEMA2				
	Anàlisis de diferents models de pèndul invertit.	29-Mar-12	15	2-Jun-12
	Instal·lació soft. latex i apendre el funcionamen	20-Jun-12	3	20-Jun-12
	Escriure tema 2	21-Jul-12	5	27-Jul-12
TEMA3				
	Recerca i adquisició d'un microcontrolador.	4-Jun-12	3	4-Jun-12
	Adaptació del microcontrolador i proves.	12-Jun-12	10	18-Jun-12
	Implementar PWM i incorporar xip l293d	19-Jun-12	3	19-Jun-12
	Adquisició mini protoboard i porta piles.	25-Jun-12	3	25-Jun-12
	Incorporació del xip MinIMU-9, anàlisi i funcion	18-Aug-12	10	18-Aug-12
	Incorporació sensor IR+ codi.	26-Sep-12	5	1-Oct-12
	Rectificacons funcionament software Arduino.	26-Sep-12	3	26-Sep-12
	Escriure tema 3	7-Oct-12	4	8-Oct-12
TEMA4				
	Documentació i obtenció del model del pèndul i	14-Oct-12	11	18-Oct-12
	Extreure el model dels motors.	20-Oct-12	1.3	20-Oct-12
	Obtenció bateria nova.	21-Oct-12	2.3	21-Oct-12
	Procés d'adaptació de dades (MinIMU-9) a grau	22-Oct-12	3	22-Oct-12
	Disseny controlador obtenció de pols polinomia	25-Oct-12	3	25-Oct-12
	Simulació sistema+controlador per pols polinon	29-Oct-12	5	29-Oct-12
	Compra nou sensor IR.	1-Nov-12	2.3	1-Nov-12
	Instal·lació del sensor IR+ nou model motors.	5-Nov-12	2.3	5-Nov-12
	Nou model planta+simulacions+criteri d'estabili	5-Nov-12	2.3	5-Nov-12
	Escriure tema 4.	6-Nov-12	3	6-Nov-12
TEMA5				
	Disseny d'un controlador PID.	7-Nov-12	11	11-Nov-12
	Disseny d'un controlador LQR.	15-Nov-12	6	17-Nov-12
	Observabilitat i Controlabilitat.	18-Nov-12	3.3	19-Nov-12
	Control amb Arduino.	22-Nov-12	4	23-Nov-12
	Correció d'errates.	24-Nov-12	3	24-Nov-12
	Estudi i simulació de l'observador.	1-Dec-12	8	2-Dec-12
TEMA6				
	Obtenció senyal de control 'u'.	3-Dec-12	3	4-Dec-12
	Disseny algoritme i sim.s amb diferents c.i.	5-Dec-12	10	10-Dec-12
	Escriure tema 6	12-Dec-12	4	14-Dec-12
Anexos			12	
	Annexos.	15-Dec-12	12	19-Dec-12
TEMA7				
	Disseny algoritme cas real	25-Nov-12	5	1-Dec-12
	Anàlisi de l'algoritme	2-Dec-12	5	03-Dec-12
	Proves	4-Dec-12	25	24-Dec-12
TEMA8				
	Conclusions	25-Dec-12	8	29-Dec-12
correccions		2-Jan-13	15	25-Jan-13

Tema	Data inici	Duració dies	Data final	Duració hores
Tema1	26-Nov-12	2	27-Nov-12	5
Tema2	29-Mar-12	121	27-Jul-12	23
Tema3	4-Jun-12	127	8-Oct-12	41
Tema4	14-Oct-12	22	6-Nov-12	35.5
Tema5	7-Nov-12	25	2-Dec-12	35.3
Tema6	3-Dec-12	12	14-Dec-12	17
Tema7	25-Nov-12	29	24-Dec-12	35
Tema8	25-Dec-12	5	29-Dec-12	8
Anexos	15-Dec-12	5	19-Dec-12	12
correccions	2-Jan-13	20	25-Jan-13	15

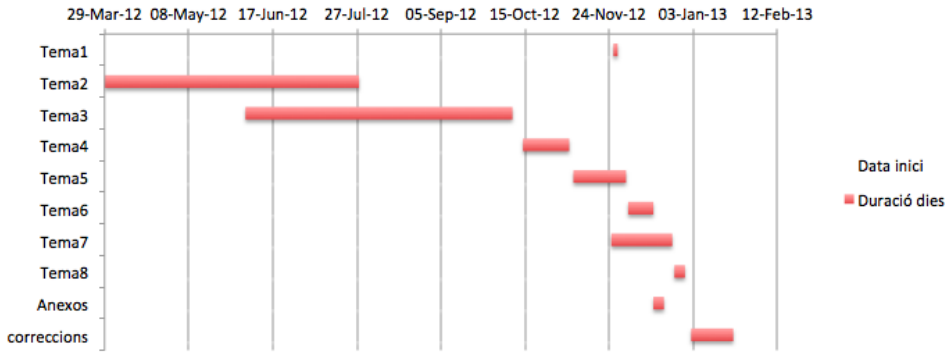


Figura G.3: *Diagrama de Gantt planificació final*

Total hores	individuals	226.8
Total hores	reunions	34.1
	TOTAL HORE	260.9

Figura G.4: *Total hores dedicades*

Resum

Aquest projecte consisteix en l'estudi del problema clàssic del pèndul invertit en l'àrea de control. Per tal de poder realitzar aquest estudi s'ha construït un prototip que simuli el comportament d'un pèndul invertit. Seguidament es dissenyen uns controladors PID i LQR per aquest prototip. Finalment s'escull el controlador LQR, que és per al qual es realitzen les simulacions i es programa el prototip real.

Resumen

Este proyecto consiste en el estudio del problema clásico del péndulo invertido en el área de control. Para poder realizar este estudio se construye un prototipo de péndulo que simule su comportamiento. Seguidamente se diseñan unos controladores PID y LQR pare este prototipo. Finalmente se elige el controlador LQR, que es para el qual se hacen las simulaciones y se programa el prototipo real.

Abstract

This project consists on the study of the classic inverted pendulum problem in the control area. To implement this study, a prototype which simulates this problem is build. Subsequently, several possible controls e.g.(PID,LQR) are designed and simulated with Matlab. Lastly, the chosen control is LQR, which is programmed in the real prototype.

Firmat: Xavier Jordà Múrria

Bellaterra, 7 de Gener de 2013